CALL IN WITHOUT LOCK-IN
Get connected with the Jitsi
videoconferencing tool

SPEED UP YOUR SYSTEM
HOT TWEAKS FOR
A FASTER LINUX

PRO
LINUX
MAGAZINE

LINUX
MAGAZINE
PRO

ISSUE 238 – SEPTEMBER 2020

SPEED UP YOUR SYSTEM
Hot tweaks for a faster Linux

PSI
Discover the kernel's cool new
performance monitoring feature

Easy Tips for Optimizing
Shell Scripts

Repology
Search 120 package
repositories at once

Steampunk Laptop
A little copper tubing
and a Raspberry Pi

Scary Tech
This Halloween vending machine
dispenses chocolate treats

LINUXVOICE

FOSSPicks
• Ardour 6.0
• Celestia
• Vitomu Audio Ripper

• maddog – IoT and resource consumption
• Record Terminal Sessions with Asciinema
• Social Proof – A new approach to
marketing Linux

Tutorial
• UltraScreen
Internet-Ready
Remote Desktop

LINUX NEW MEDIA
The Pulse of Open Source
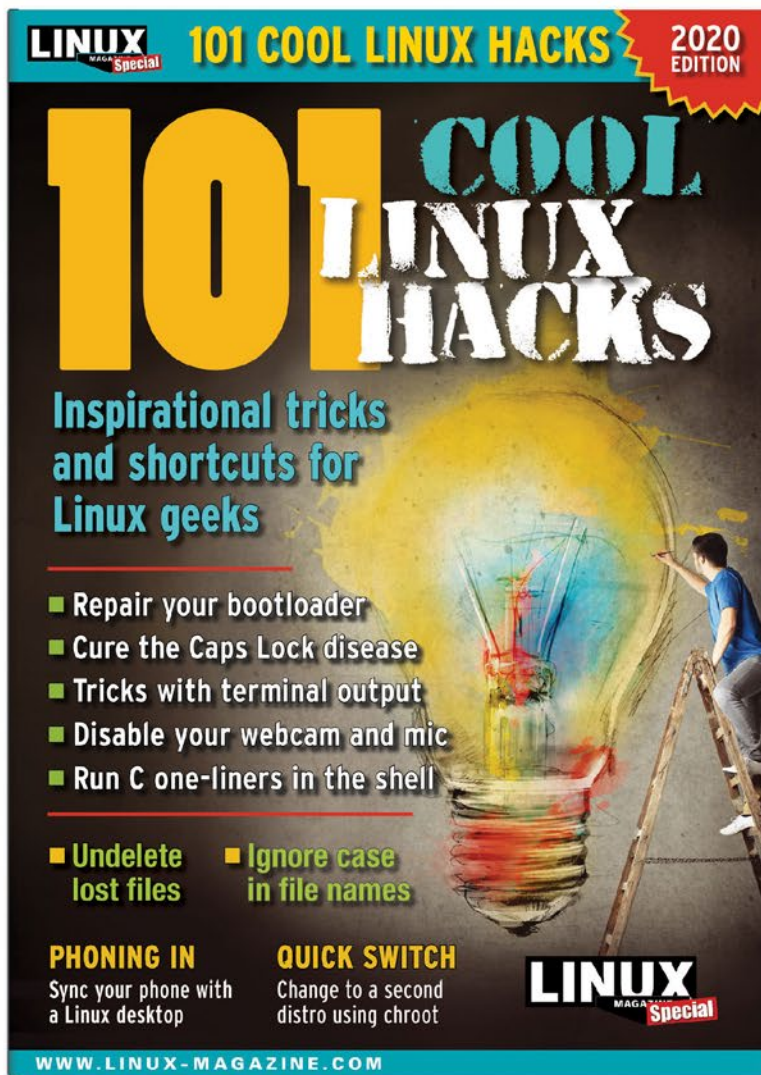
PROLIFERATION

Dear Reader,

Name an open source license. Most readers of this magazine would think of the GNU Public License (GPL), but other licenses also occupy a portion of community attention. For instance, most seasoned Linux users are aware of the BSD license and the Apache License, as well as the Mozilla Public License and the Lesser GPL. Each license serves a slightly different need, so each license is slightly different.

A menu of different open source licenses is actually healthy, because it gives users flexibility in choosing a license based on the project's priorities and needs. But seriously, how big does this menu really need to be? The Open Source Initiative (OSI) lists 103 licenses that meet the Open Source Definition [1]. 17 of the licenses on the list are marked as retired or superseded, but that still leaves 86 active licenses that all qualify as open source. As you can probably guess, most of these licenses are hardly ever used, but they still meet the definition, so they found their way onto the list.

Open source scions have been concerned for some time about the dangers of license proliferation. Too many licenses adds confusion and detracts from the clarity and unity of purpose that is so fundamental to the spirit of open source. It also makes it more difficult for an average user to understand the rules for using a software product. Linux users mostly know what the GPLv2 is, and when they read that an application is licensed under the GPLv2, they know what that means. They don't necessarily know what the Sybase Open Watcom Public License 1.0 is and why it is different, so a user who is conscientious about learning the rules that apply to the software needs to spend more time pouring over the

legal fine print. Ultimately, if too many unfamiliar licenses are in circulation, using an open source product would lead to the same kind of "OK…whatever" absent acquiescence that is so common in the closed-source world, with users agreeing to EULAs they haven't even read.

The Open Source Initiative has its eye on the problem of license proliferation, and they are taking steps to address it. First of all, they publish a list of licenses they call "popular and widely-used or with strong communities" and encourage developers to use one of these licenses as a first option (see the box entitled "OSI's A List"). To encourage reuse, they insist that all OSI-approved licenses are clearly written and don't include language that limits the use to a single project or industry. In spite of these efforts, the number of OSI-approved licenses has roughly doubled from the approved list of 50 that existed in 2004.

Is all this heading the way they want it to go? Not so much. OSI announced this month that they will convene a License Listing working group to study the licensing process and make recommendations about any changes that would lead to a better and more orderly outcome. According to the OSI website, "The mission of the License List working group is to find ways to ensure that the OSI's license list is appropriately comprehensive while also continuing to encourage the use of a smaller set of well-known, well-understood licenses" [2]. The group will look at the criteria for approving licenses and will even consider whether there is a need for another process to de-certify a license that is no longer needed or is too similar to another.

Interestingly, OSI also raises the question of whether a preoccupation with license proliferation has led to some worthy licenses *not* making the list. One of the topics the group will study is whether "…a desire to discourage license proliferation may have prevented some suitable licenses from obtaining OSI approval."

OSI says the License List working group will not consider changes to the official Open Source Definition, but everything else about the approval process is on the table. Open source software has way more stature and visibility than it did back in 1998 when the OSI was founded. Open source licensing is actively desired by many projects, which should strengthen the hand of the OSI when cleaning up the license list and defining a more practical approach. With any luck, the end result should be a more sensible process and a comprehensive list of core licenses that will leave fewer reasons for future proliferation. ∎∎∎

OSI's A List

- Apache License, 2.0

- BSD 3-Clause "New" or "Revised" license

- BSD 2-Clause "Simplified" or "FreeBSD" license

- GNU General Public License (GPL version 2)

- GNU Library or "Lesser" General Public License (LGPL version 2)

- MIT license

- Mozilla Public License 1.1 (MPL)

- Common Development and Distribution License

- Eclipse Public License

Info

[1] OSI-approved Licenses:
https://opensource.org/licenses/alphabetical

[2] License List Working Group:
https://wiki.opensource.org/bin/Working+Groups+%26+Incubator+Projects/License+List+Working+Group/

Joe Casad,
Editor in Chief

LINUX MAGAZINE

WHAT'S INSIDE

Your Linux experience goes much more smoothly if your system is running at peak performance. This month we focus on some timely tuning techniques, including the kernel's new Pressure Stall Information (PSI) feature.

Also inside:

- **Jitsi** – a reliable solution for open source video conferencing (page 44).
- **Repology** – this useful web app serves as a universal search engine for Linux packages (page 50).

At MakerSpace, we get ready for Halloween, and LinuxVoice offers some new insights on marketing Linux.

SERVICE

NEWS

It's been a rough couple of years for the German-based SUSE and its flagship open source project, openSUSE. *Linux Magazine* talks to Dr. Gerald Pfeifer about where openSUSE is going and its relationship to SUSE.

COVER STORIES

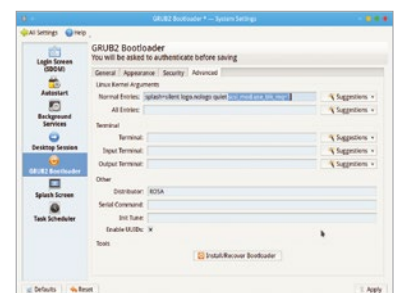Pressure Stall Information (PSI) is a new feature that gives users a better view of resource contention.

Shell scripts are often written for simplicity rather than efficiency. A closer look at the processes can lead to easy optimization.

If you are looking for ways to speed up your Linux, consider this collection of curated performance tweaks.

TWO TERRIFIC DISTROS
DOUBLE-SIDED DVD!

SEE PAGE 6 FOR DETAILS

LINUXVOICE

openSUSE 15.2 and Bodhi Linux 5.1
Two Terrific Distros on a Double-Sided DVD!

OpenSUSE is one of the most popular Linux distributions. It is available in two versions: Leap, the stable version intended for average users and system administrators, and Tumbleweed, a rolling release for experimenters and those who want the very latest in hardware. Leap shares a common codebase with the commercial SUSE Linux Enterprise distribution, while Tumbleweed is developed in Factory, openSUSE's main development codebase, before release to help assure relative stability. All these projects are closely associated with each other.

In the past, the SUSE family of distributions was known for its contributions to KDE. Today, it is perhaps better known for its support of the Wayland graphics manager in the Gnome desktop environment. It also is known for a variety of unique tools, including the YaST Control Center, as well as AutoYaST, which provides automatic installations, and WebYaST, a browser interface. The SUSE family is also known for the ZYpp package manager and for the use of delta RPMs, which contain only the differences between old and new packages of the same name, a feature that speeds software installation. In addition, the SUSE family has developed its Build Services for packaging applications. It has experimented with automatic testing to improve software quality, as well as with the use of Flatpaks in one release variant. Although in many ways a mainstream distribution, openSUSE is at the forefront of innovation among distributions.

Bodhi Linux is a lesser known distribution with a cult following that deserves greater recognition. It is best known for its Moksha desktop, which is based on the Enlightenment window manager. Moksha provides a fast, lightweight desktop while offering a broad range of customizations and utilities ideal for systems with limited hardware. Unlike many current distributions, Bodhi also continues to release a 32-bit version, making it an ideal choice for keeping older systems alive. In addition, Bodhi supports AppImages, an early form of universal packages similar to Flatpak and Snap.

Bodhi 5.1 is the first release in almost two years, due partly to a change in project management and members. Former users will find the latest release lives up to its predecessors. A few apps have changed – for instance, the ePad text editor has been replaced by Leafpad, while a Bodhi-branded version of Gnome Web replaces the Midori web browser. Because part of Bodhi's philosophy is to preinstall a minimum of packages, these changes have a minimal effect. The dock options are especially extensive and, along with the wide array of themes, allow Bodhi to resemble other operating systems or a desktop based on your personal preferences.

A side benefit of Bodhi is that it is based on a Long Term Support release of Ubuntu. If Bodhi's own documentation does not help you with technical matters, Ubuntu's support forum and online help may.

Defective discs will be replaced.
Please send an email to subs@linux-magazine.com.

NEWS

Updates on technologies, trends, and tools

THIS MONTH'S NEWS

Purism Launches the Librem Mini

For anyone looking to deploy small form factor PCs and wanting them powered by Linux, Purism might have what you're looking for. The Librem Mini (*https://puri.sm/ products/librem-mini/*) is a tiny device that packs plenty of features. The form factor is smaller than a Mac Mini, bigger than a Raspberry Pi, and includes everything you need to work with Linux.

The specs for the device are: CPU – Quad-Core i7 (Whiskey Lake) with 4 cores, 8 threads, and up to 4.6GHz; RAM – Up to 64GB; Storage – 250GB SATA M.2; Graphics – Intel UHD 620; and Networking – Gigabit ethernet (with optional WiFi).

On the front of the device you'll find a power button, a headphone jack, and 4 USB 3.0 ports. On the back of the device, you'll find a standard ethernet jack, two USB 2.0 ports, one USB-C port, a 4K HDMI 2.0 port, and a Display port.

Inside, however, is where the Librem Mini shines. Purism has boosted security by disabling and neutralizing the Intel Management Engine, using Coreboot-based OSS firmware and boot process validation (with the help of PureBoot and a Librem Key), and (of course) the Librem version of Linux, Pure OS.

The cost of the Librem Mini starts at $699 for the 8GB model, but can be specced out to nearly $3,000. When the units were listed, Purism said they wouldn't ship until they received $50,000 worth of pre-orders. That goal was met on April 5, 2020, so the units are now available for shipping from the Purism Store (*https://shop.puri.sm/shop/ librem-mini/*).

Image © https://shop.puri.sm/shop/librem-mini/

openSUSE Leap 15.2 Adds AI and Machine Learning

For openSUSE users, there's some very exciting news for the release of the latest iteration, 15.2. This new take on the platform includes several new packages in the mix that add both artificial intelligence (AI) and machine learning. The packages that bring these two new features include Tensorflow (a framework for deep learning), PyTorch (a machine learning library), and ONNX (an open format for machine learning models that provides interoperability in the AI tool space).

Other new features to openSUSE Leap 15.2 include Grafana and Prometheus packages for analytical experts. The latest release also has a real time kernel available (which is beneficial for systems that need to process tasks in a given time, else the task may fail). For container administrators, there's finally an official Kubernetes package available. On top of the official Kubernetes package, the Helm Kubernetes package manager has been added to give a boost to container orchestration capabilities. To help make container orchestration more efficient, Container Runtime Inter-

face using Open Container Initiative conformant runtimes has been included and serves as a lightweight alternative to the Docker runtime. For network and security admins, Cilium can aid in transparently securing network connectivity and load balancing between application containers.

openSUSE Leap 15.2 also offers Server and Transactional Server system roles. The Server role includes a small set of packages best suited for GUI-less servers, while the Transactional Server role is similar to the Server system role, only it makes use of a read-only filesystem for atomic and automatic updates.

Get your copy of openSUSE Leap 15.2 from the official download page (*https://software.opensuse.org/distributions/leap/15_2*).

Google's Nearby Sharing Could Work with Linux

For Linux users, one of the big frustrations with Android is transferring files to and from a mobile device. You could certainly set up an SMB share and install an Android file manager that includes Samba connectivity, or configure an FTP or sFTP server on your Linux machine and use a supporting app on Android.

That necessity could be coming to an end.

If the rumors are true, Android will soon be bringing a new feature to the mobile platform that will allow you to easily share things to nearby devices. That feature is aptly named, Nearby Sharing. Originally this feature was to be used for sharing links, photos, and more between Android devices. However, Google has brought the feature to the latest Chrome OS Canary builds. And although the feature has yet to actually do anything, the description of Nearby Sharing for Android is the same found in the Chrome browser_features.cc (*https://chromium.googlesource.com/chromium/src/+/23de21 d5287a72ef254f486770ac791d66eee3f1/chrome/browser/browser_features.cc#17*) file, which indicates the feature will be coming to the Chrome browser on Windows, macOS, and Linux.

Yes, that mean Nearby Sharing wouldn't technically work natively on Linux, but requires the use of Chrome on the open source platform. Even so, it would be an important step forward for any Linux user who depends on Android and needs an efficient way to transfer files back and forth between Google's mobile OS and their desktop of choice.

Read the original commit for the Chrome browser file here: *https://chromium-review.googlesource.com/c/chromium/src/+/2133053*.

System76 Launches Ryzen-Powered Laptop

System76 is never one to rest on their reputation or become complacent in an incredibly competitive field. This has never been more apparent than with their constant push to bring more and more powerful systems to consumers. Case in point, the new Serval WS laptop (*https://system76.com/laptops/serval*). This Linux workhorse includes a powerful AMD Ryzen CPU, along with all the other bells and whistles you'd expect in a flagship laptop.

But the highlight of the Serval WS has to be the processor. System76 offers three different configurations for this laptop: Ryzen 3600; Ryzen 3700; and Ryzen 3900.

Of those three options, it will be the Ryzen 3900 that will garner the most attention, as it's a CPU that includes 12 cores and 24 threads.

Consumers can also choose between a GTX 1660 Ti or RTX 2070 GPU, for graphics intensive workloads. Other features of the Serval WS include an RGB keyboard, 1080p webcam, and plenty of ports for connectivity. The laptop can be purchased

MORE ONLINE

Linux Magazine
www.linux-magazine.com

ADMIN HPC
http://www.admin-magazine.com/HPC/

Caching with CacheFS
• Jeff Layton
For read-heavy workloads, CacheFS is a great caching mechanism for NFS and AFS.

ADMIN Online
http://www.admin-magazine.com/

WiFi 6 in the Enterprise
• Mathias Hein
The 802.11ax WiFi 6 standard uses LTE technology to solve the congestion problems experienced with older standards.

Managing Bufferbloat
• Mathias Hein
Bufferbloat impedes TCP/IP traffic and makes life difficult, especially for real-time applications like voice or video.

An overview of the Citadel BBS
• Rubén Llorente
Citadel is a free, compact bulletin board system that integrates email, calendar, address book, and forum functions for administrators and hobbyists that need a groupware solution for a small organization.

with either a 15" 1080p or 17" 4K display. The Serval WS supports up to 4TB of NVMe SSD storage and up to 64GB of memory.

The base model starts at $1,299 and can be configured to a whopping $5,000. Like all System76 machines, the Serval WS comes preloaded with their in-house Linux distribution, Pop!_OS (*https://system76.com/pop*).

Fedora 33 Desktop Defaults to Btrfs

Although the release of Fedora 33 is a few months away, there has one been one announcement that has turned a few heads. Said announcement is the shift to the btrfs file system as the default for the desktop. All desktop editions of Fedora 33, including spins and labs will be included in this change.

But why have the Fedora developers opted to make this change? The developers see btrfs solving a number of problems, such as users running out of free space, preventing data corruption caused by flaky storage devices, poor responsiveness when under pressure, file system resizing, and complex storage setups.

Btrfs on Fedora 33 will include all of the default features found in the stable version of the file system. And with btrfs having been well tested for over 7 years, the developers are confident this change will go smoothly. To this point, the Fedora developers have said:

"For laptop and workstation installs of Fedora, we want to provide file system features to users in a transparent fashion. We want to add new features, while reducing the amount of expertise needed to deal with situations like running out of disk space. Btrfs is well adapted to this role by design philosophy, let's make it the default."

This change will not have any impact on Red Hat Enterprise Linux.

Original source: *https://fedoraproject.org/wiki/Changes/BtrfsByDefault*

SUSE Acquires Rancher Labs

Rancher Labs is a privately held company that provides a Kubernetes management platform. SUSE is one of the largest independent open source companies. Put these two together and you have the makings for an incredible moment in IT history.

Melissa Di Donato, SUSE CEO said of this definitive agreement. "Only the combination of SUSE and Rancher will have the depth of a globally supported and 100% true open source portfolio, including cloud native technologies, to help our customers seamlessly innovate across their business from the edge to the core to the cloud."

Unlike other solutions, the SUSE/Rancher Labs combination will remain fully open source. And with cloud-native application adoption continuing to grow exponentially, this coming together of a powerhouse and an up-and-coming star should be a game-changer. With SUSE being a global leader in Enterprise Linux, Edge Computing, and AI, and Rancher already being established as a leader in Kubernetes container management, this acquisition will deliver "computing everywhere" with the latest AI and containerized workloads.

Of this combination, Sheng Liang, Rancher CEO, says "Our leading Kubernetes platform with SUSE's broad open source software solutions creates a powerful combination, enabling IT and Operations leaders worldwide to best meet the needs of their customers wherever they are on their digital transformation journey from the data center to cloud to edge."

The acquisition of Rancher is the first step in SUSE's planned expansion strategy since becoming a fully independent software company in March 2019.

It is rumored that SUSE is paying at least $600 million to acquire Rancher labs. The deal should close around October 2020.

Source: *https://rancher.com/blog/2020/suse-to-acquire-rancher/*

**Get the latest news in your inbox every two weeks**

Subscribe FREE to Linux Update

bit.ly/Linux-Update

LINUX UPDATE

Need more Linux?

Our free Linux Update newsletter delivers insightful articles and tech tips to your mailbox twice a month. You'll discover:

- Original articles on real-world Linux

- Linux news

- Tips on Bash scripting and other advanced techniques

- Discounts and special offers available only to newsletter subscribers
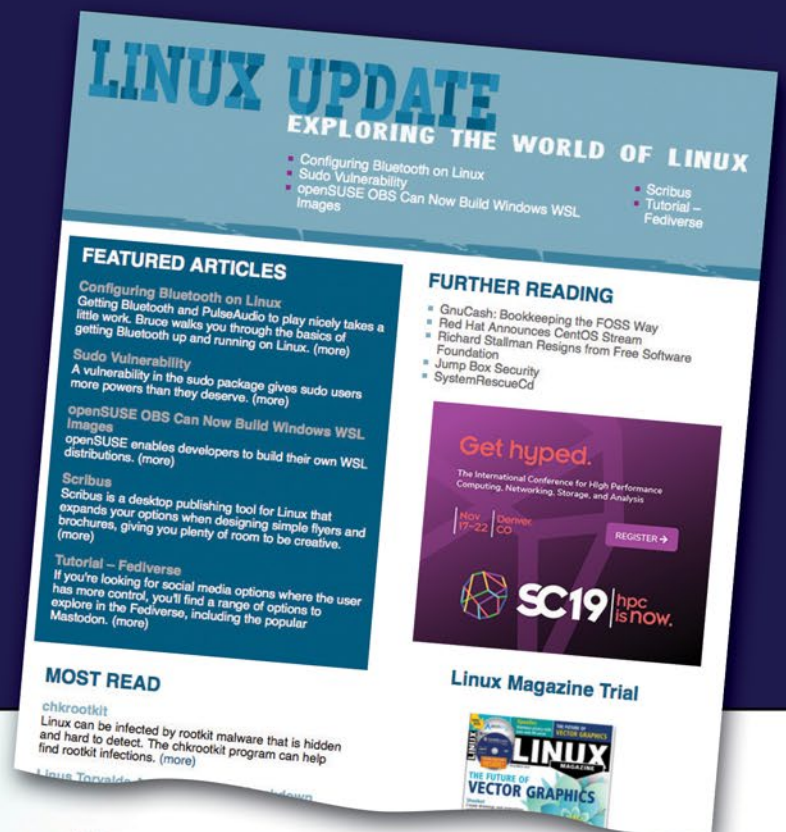
bit.ly/Linux-Update

Zack's Kernel News

Chronicler Zack Brown reports on the latest news, views, dilemmas, and developments within the Linux kernel community.

By Zack Brown

Author

The Linux kernel mailing list comprises the core of Linux development activities. Traffic volumes are immense, often reaching 10,000 messages in a week, and keeping up to date with the entire scope of development is a virtually impossible task for one person. One of the few brave souls to take on this task is **Zack Brown**.

Improving Memory Management

SeongJae Park wanted to improve Linux's memory management. To do this, he wanted to implement finer-grained data access tracking. Then, using that data, Linux would have a better idea of how to move data around in memory and which data to swap out to disk.

The problem was that finer-grained tracking would take CPU time that the system might otherwise use for executing user code, so there'd be a slowdown. On the other hand, the improved memory management made possible by SeongJae's patch would speed up the system – potentially enough to justify the hit taken by gathering the data.

SeongJae pointed out that there were already patches to take advantage of the kind of fine-grained data he wanted to track, but none of those patches had been merged into the Linux kernel because of the lack of that kind of tracking.

So, he announced DAMON, a data access monitoring subsystem that he claimed was accurate, lightweight, and scalable to large systems. He had also implemented it as a standalone kernel module, with a general-purpose API to allow not just the kernel to track memory access, but also allowed user code to do it too.

SeongJae posted some numbers illustrating the speed and effectiveness of the tracking system and said he had a lot more stuff planned for the future. He wanted to automate a lot more of DAMON's action at run time and support more than just RAM – he felt his code could quickly be made to support tracking the page cache, NUMA nodes, specific files, and whole block devices.

He anticipated the potential criticism that DAMON should be implemented as part of the `perf` performance analysis system, saying that while `perf` was restricted to use by the Linux kernel only, DAMON's API supported user programs as well. So to that extent, the two tools offered different capabilities.

There was no discussion of whether or not to accept SeongJae's patches into the kernel, but there was a bit of a technical discussion between him and Jonathan Cameron. Jonathan noticed that Seong-Jae's code could merge an arbitrary number of regions of memory into a single region, while splitting regions of memory would only halve each region, thus doubling the total number of regions. He felt that since any number of regions could merge into one, but a single region could not split into any number of smaller regions, this created an imbalance that could cause the system to end up with a very small number of regions in the normal case.

The problem with that is that a program could end up allocating a memory region that was much larger than it actually needed, leaving only a small amount of memory available for other allocations.

Jonathan proposed changing DAMON's region splitting algorithm to split regions twice if certain conditions were met, thus ensuring that there would remain a useful number of regions for the system's needs.

SeongJae agreed that Jonathan's proposal would work. But he also pointed out that DAMON's existing algorithm would accomplish the same thing, but more slowly. And Jonathan's fix would complicate the code somewhat. Seong-Jae wanted to wait until users had time to really expose any serious bugs and only then consider incremental improvements like Jonathan's suggestion.

But Jonathan didn't agree that the current code would accomplish the same thing, only more slowly. He felt the current code would not resolve the splitting/merging conflict at all. He posted an example, playing out the logic of Seong-Jae's code, showing how it wouldn't accomplish the same thing as Jonathan's suggestion, even at slower speeds.

And this made sense to SeongJae. The problem, as Jonathan had pointed out, was that if a "hot region" of memory was right in the middle of a given

region, it wouldn't be noticed, because the splitting process would split that hot region itself.

So, Jonathan admitted this was a "pathological case," and SeongJae accepted the patch to fix it.

I find discussions like this one interesting, because it illustrates the open source saying, "given enough eyeballs, all bugs are shallow." Jonathan identified a true bug, albeit one that was extremely unlikely to ever be a problem. SeongJae didn't see it at first, but finally together the two of them were able to produce an improved patch. It's a fun aspect of the open source development model, as is the fact that we all get to watch.

Simplifying(ish) the Kernel Build System

The kernel build system uses GNU Make at its core, but it does not constrain itself to using only GNU Make's standard features. There are a lot of ifs, ands, and buts that go into deciding which code to build into the kernel versus as a module and which options to pass to GCC. It's not your ordinary build system.

Recently, Saeed Mahameed posted a patch to update the build system to handle certain dependencies in a clearer way. He wanted to address cases where one kernel feature, built into the binary, needed to be able to reach another feature that was only built as a loadable module. Technically, both could be considered part of the kernel, but there would still be a disconnect if the one feature needs to know specific addresses into the other feature that doesn't exist before run time.

Saeed's patch introduced the `uses` keyword into the build system, so if one feature "uses" another, the build system will know to make sure that the addresses are available.

Essentially, the `uses` keyword provides an intuitive way to express this type of dependency. Then during processing, the build system replaces it with a freakishly incomprehensible logic statement that means exactly the same thing and which can then be interpreted correctly by the build system.

Arnd Bergmann was happy to see Saeed's patch and confirmed that it worked on his system, for the most part. But Arnd was able to crash the

build system with a more complex task, so he submitted a bug report. This prompted Saeed to produce a new version of his patch.

However, Masahiro Yamada didn't like Saeed's patch – or at least, felt it was not necessary. He pointed out that adding the `uses` keyword didn't actually provide any new functionality; it simply offered a simpler way to do the exact same thing that had been standard before.

Masahiro said, "It is true that it _hides_ the problems and makes the _surface_ cleaner at best, but the internal will be more complicated." He added that anyone trying to actually understand the `uses` keyword would still have to delve into the underlying nightmare logic. And for users who failed to understand the keyword, he feared they would start using `uses` all over the place, even where it wasn't truly appropriate. He concluded, "I do not want to extend Kconfig for the iffy syntax sugar."

You'll notice I've avoided actually showing any of the underlying logic until now. But the type of statement being replaced by `uses` boils down to "X or not X," which logically might be seen to mean nothing at all. But Nicolas Pitre said it was the equivalent of "depends on X if X," which Masahiro did not feel was a big improvement. In any case, in the language of kernel configuration, he said, the "depends on" construct didn't support taking a conditional like that.

The real problem may be that the logic necessary to decide what to do simply doesn't lend itself to clear or simple statements. If one kernel feature will take advantage of another kernel feature if and only if that second kernel feature is compiled into the kernel, but it will fail gracefully if the second feature is completely unavailable, just so long as the second feature is not actually loaded in as a stand-alone module, that's not necessarily something that's easy to express with a clear and simple keyword.

And so, in the normal course of events, side effects like code being "reachable" come to represent one or another complicated situation. Then those side effects are codified into keywords and paired up with other keywords into apparently logically connected AND/NOT/OR/XOR statements

that do what's needed, but God help anyone who really needs to know what's going on.

Thus, the kernel configuration system continues to grow.

In the current debate, one of the main points at issue was how well users (i.e., developers of kernel modules) could be made to understand the keywords that would express their modules' dependencies on other parts of the system. Or at least, how easily they could figure out which configuration text to cut and paste from some other module into their own.

At one point, Nicolas offered one example of this problem when he remarked to Saeed, "I don't dispute your argument for having a new keyword. But the most difficult part as Arnd said is to find it. You cannot pretend that 'optional FOO' is clear when it actually imposes a restriction when FOO = m. Try to justify to people why they cannot select y because of this 'optional' thing."

There's a real benefit to be gained. If users can't figure out how to express their dependencies correctly, they will end up depending on more than they need to, just in order to err on the side of caution. This then increases kernel build times and possibly bloats the compiled binary with unnecessary code.

Nicolas actually went on to propose his own solution. He blurted out, "saying that 'this is weird but it is described in the documentation' is not good enough. We must make things clear in the first place." And went on to say:

"This is really a conditional dependency. That's all this is about. So why not simply making it so rather than fooling ourselves? All that is required is an extension that would allow:

```
depends on (expression) if (expression)
```

This construct should be obvious even without reading the doc, is already used extensively for other things already, and is flexible enough to cover all sort of cases in addition to this particular one."

At this point Jani Nikula said he agreed with Nicolas's proposal and asked for an implementation. Nicolas sent in a patch, and Randy Dunlap said he preferred this over other proposed solutions.

The conversation then descended into one particular special case that various people wanted to fold into Nicolas's patch. So the conversation petered out at around this point.

When I see debates like this, where it's obvious that everyone is struggling to make sense out of the nearly nonsensical, it gives me a strong sense of appreciation for the people choosing to bang their heads against these particular rocks. Eventually the kernel build system will be clean and beautiful and easy to use, but that day will only come because of the psychotic determination of obsessive lunatics like Nicolas, Saeed, and the rest.

Detecting Firmware Crashes

Luis Chamberlain wanted to make it easier for Linux to handle bad firmware. He said, "Device driver firmware can crash, and sometimes, this can leave your system in a state which makes the device or subsystem completely useless. Detecting this by inspecting /proc/sys/kernel/tainted instead of scraping some magical words from the kernel log, which is driver specific, is much easier. So instead provide a helper which lets drivers annotate this."

Luis's helper function actually does the magical scraping itself and simply puts the results into the `/proc` file. So the goal is to save everyone else from

having to do it. But this means that Luis's code has to know how to scrape the truth out of every single device driver in the kernel.

As of his post, he had covered the device drivers that have names starting with Q.

He pointed out that one of the motivations for this work was not only to make it easier for Linux to detect these crashed firmwares, but also to make it easier to support users in general, by ruling out firmware as the cause of various bug reports.

Kees Cook loved Luis's code. But instead of just a single patch, he suggested splitting the entire project into separate patches for each maintainer – that way each patch could be reviewed by the appropriate people, instead of potentially getting hung up waiting for some maintainers, while others responded more quickly.

Daniel Vetter and Rafael Aquini agreed with splitting up the patch, and Luis said he'd give it a shot.

Steven Rostedt also said he liked Luis's work.

So there seemed to be universal approval. Aside from practical changes like splitting the patch, there were some technical comments, but no one objected to the project itself or the implementation.

The hardest part of the project, ultimately, may just be maintaining all the scraping code, so that crash detection continues to be accurate as each driver continues to develop. But it's possible that maintaining crash detection will become just another part of each maintainer's responsibilities, so that Linus may start refusing driver updates from maintainers who don't maintain good corresponding crash detection code. But that's just a guess. ∎∎∎

The openSUSE/SUSE relationship

Open Sesame

It's been a rough couple of years laced with uncertainty for the German-based SUSE and its flagship open source project, openSUSE. *Linux Magazine* talks to Dr. Gerald Pfeifer about where openSUSE is going and its relationship to SUSE. *By Mayank Sharma*

OpenSUSE [1] was once one of the leading open source RPM-based distributions and one of the handful of distributions that operated under the auspices of a for-profit corporation. While SUSE had been marketing Linux for the enterprise since the '90s, the openSUSE project began in 2005, a couple of years after the company's acquisition by Novell. The last decade, however, has been very turbulent for SUSE, which has changed ownership several times.

At the Open Source Summit in Lyon, France, in 2019, I caught up with Dr. Gerald Pfeifer [2], who is the CTO at SUSE and chair of the openSUSE board. Despite the multiple changes of ownership and the resulting apprehensions from the community, Pfeifer assures that the openSUSE project continues unabated. Pfeifer has been actively involved in the open source community for several years, contributing to various open source projects, including GCC, Wine, and FreeBSD. In our conversation, he explains the current status of the openSUSE project and how its relationship with SUSE's Enterprise offering is a win-win for both the company and the project.

Linux Magazine: Let's start with your role at SUSE?

Gerald Pfeifer: I wear two hats. For example, at this event, I'm CTO at SUSE based in EMEA (Europe, the Middle East, and Africa). There's two of us: one based in North America, me based in EMEA. And since August, I also am the chairperson of the openSUSE Board.

LM: It's been a turbulent few years for SUSE with all the acquisitions. Where do things stand now? Who owns what, and what does it mean for openSUSE?

GP: SUSE now is wholly owned by EQT, a Swedish investment fund. That was announced in July of 2018. And then in spring 2019, mid-March, is when SUSE became really independent. The process of becoming independent kept us busy.

LM: How does this change of ownership affect openSUSE?

GP: I have talked with people in the openSUSE communities, and I've heard that some are a little concerned about the change of ownership. But practically, so far there hasn't been a change for the openSUSE project. Full stop. And there is no concrete plan of changing anything around openSUSE, driven from a SUSE perspective or from our owners. When I became chair of the board in August 2019, I started to look into the situation from both sides and had many conversations. One thing I found is that there is an opportunity for SUSE to communicate better with openSUSE our intentions, our commitments, and our ongoing commitments. For example, Melissa Di Donato, our CEO of SUSE, has sent a message to the openSUSE Asia Summit that happened in Bali first weekend in October re-emphasizing the importance of openSUSE for SUSE.

Equally what I found is many of us working for SUSE, including myself even before that change, have a foot in both openSUSE and SUSE. Even then, I and others at SUSE may not fully see all the good things that happen in the open-

SUSE communities. It actually could be a good thing if, in both directions, we were to share more – share more about the positives, share more about the positive intentions. This creates additional synergy effects.

One thing that I found is that at openSUSE we're not terribly good at marketing some of our successes and some of our accomplishments towards the world. And that's something I have started to encourage, and I do want to make this one of my contributions.

LM: One thing that you don't communicate very well is that both Leap and Tumbleweed have a relationship with SUSE Linux Enterprise (SLE). Could you please explain that relationship?

GP: OpenSUSE Tumbleweed [3] is a rich, rolling Linux distribution with lots of content, like probably a dozen desktop environments, that you can select as a user. Different components are updated, refreshed, and refined on an ongoing basis. Think of it as a big stream: At nearly any point in time, water is taken out, and fresh water is coming in from upstream. There is a new Linux kernel; there is a new glibc, a new compiler, [and] all the many pieces. We have a fabulous release engineering team that makes sure this all happens very smoothly. They bundle updates and put them into sub-projects that are tested independently – and this is where we have a really cool software called openQA [4] that does a lot of automated tests. If all those tests succeed, everything is integrated, and there is a new snapshot release. This sometimes happens weekly,

sometimes twice a week, thrice a week, or so. It's a rolling release, where you essentially get updates of everything, whenever someone packages it, whenever the developers fix some of the implications. For example, you include a new compiler, and all of a sudden a number of packages don't build anymore, because the compiler became more standards compliant, and those packages were not. So you start fixing those packages, work with upstream so they fix, or we do a fix in openSUSE and share it with the upstream project.

LM: But what is the relationship with SLE?

GP: When we're thinking to do a major release of SUSE Linux Enterprise, we pick one point in the stream of openSUSE Tumbleweed and branch, and, say for instance, this is where SUSE Linux Enterprise Server 15 productization starts. Then all those components that you want in SLES 15 form their own code stream and our enterprise processes take place, which focus more on stabilization, documentation, testing, and iterating those steps without updating all the individual components.

Obviously, everything we find in that context is again pushed into the upstream projects directly and openSUSE as applicable, so it's not a one-way street. This is where openSUSE Tumbleweed feeds into a SUSE Linux Enterprise major release.

OpenSUSE Leap [5], on the other hand, combines SUSE Linux Enterprise source code and additional components coming in from openSUSE Tumbleweed. A little simplified, but think of it as a two layer cake, where the base – the Linux kernel, everything around hardware, really the fundamentals of the system – comes from SUSE Linux Enterprise, and then some of the elements higher up in the stack, often things related to the graphical user interface or additional tools, come either from there as well or from the current state of openSUSE Tumbleweed. In other words, openSUSE Leap is downstream of both SUSE Linux Enterprise and Tumbleweed, combining these two.

LM: Why did you choose this approach?

GP: As I said, openSUSE Tumbleweed is actually surprisingly stable, very func-

tional, given that it's a rolling release. I'm using it on my notebook. It's the only operating system currently on my notebook, and I'm quite satisfied.

One of the drawbacks of something like Tumbleweed is, and I see this with friends who are just users, that you regularly get bigger updates. This is sometimes because there's a new version of an application, the user interface changes a little, or just the sheer amount of data that you download. Sometimes it's 300MB; if I don't update for a week or two, then it's 500MB or more, sometimes beyond a gigabyte. So that's a lot. Which is fine for me to consume or techie users, but not necessarily for the persona (as we call them in product management: the persona "partner"). You're the techie, and your partner has a notebook, and all he or she wants is [to] read email, browse the web, [or] manage pictures. So, having something that is stable and has a richer set of applications – especially for desktop users, etc. – is what we wanted to do.

And so we said, let's take the stability of SUSE Linux Enterprise and add more of this richness of the openSUSE ecosystem. Because for SUSE Linux Enterprise, one design criterion we have as a default is "best of breed." We try to focus on one of each: one web server, one desktop environment, one this or that. We deviate when there's a strong user demand. For example, we are shipping KVM and Xen, because we have users of both. But on the desktop environment, we're really focused on Gnome, and KDE is not supported as part of SLE. With openSUSE, we wanted to have more richness, and the upper layer of the cake gets a lot from the openSUSE side. This approach combines stability and richness, and the result is Leap.

LM: SUSE has recently proposed bringing the code streams of SUSE Linux Enterprise and openSUSE Leap closer together [6]. How does this proposal affect the existing relationship between openSUSE and SLE? What does it mean for users of both openSUSE and SLE?

GP: The idea behind this proposal, which I relayed on behalf of SUSE, is to strengthen the relationship; on the one hand, by sharing not just source code,

but the actual SUSE Linux Enterprise binary packages for use by openSUSE Leap, [and] on the other hand, by giving openSUSE developers more influence on the development of SUSE Linux Enterprise past the initial point when it branches from Tumbleweed.

As a practical first step, SUSE engineers are analyzing where Leap differs from SLE and adjusting the latter where possible, even if that implies extra effort now or during the life cycle. In parallel, we are also working to open up processes and tools on the enterprise side – think feature or bug tracking.

Why all this? We want to further strengthen the openSUSE project and the position of Leap as a platform for open source communities as well as industry partners. Intensifying collaboration and combining forces benefits users on both sides, and full compatibility between Leap and SUSE Linux Enterprise benefits developers who can target both at once and users who can easily migrate.

LM: SUSE has an interesting product lineup. There's SUSE Linux Enterprise Server for ARM/RaspberryPi, and you also used to sell a version of LibreOffice.

GP: When I was responsible for product management, we sunset LibreOffice as a product. And the reason is very simple. We sold LibreOffice as a Windows product, and it was the only Windows product in our portfolio. From a strategic perspective, but also from a go-to-market perspective, that just didn't work. It's something we inherited from the Novell days. That has been taken over by a partner called Collabora, and some of those team members stayed with SUSE in different areas, and some of the team actually went to Collabora and created the LibreOffice group at Collabora.

LM: Do you have some desktop offerings as well?

GP: We do. We have a SUSE Linux Enterprise Desktop and the Workstation Extension that on top of a full-fledged server product provides desktop functionality. One use case is technical workstations. For instance, you have some CAD software that is certified on the server OS, and you want to run it on your desktop.

LM: Coming to the meat of it, the enterprise back end: You are betting big on containers, especially Kubernetes. The two products that you seem to be excited about are Container as a Service (CaaS) and the SUSE Cloud Application Platform.

GP: Yeah, I'm really excited about them. If I meet someone who has not had a history with SUSE, the way I usually start describing what we do focuses on application delivery. Why have people even started to use Linux? Two reasons: One is to run some part of the infrastructure – be it a web server, be it a print server – so something which a Linux distribution can do, can provide itself. Or, and that's really what happened more and more after the early days of Linux, it is to provide the platform for stuff to run on. And originally, what that meant is, you had an operating system, and you ran one application on that.

But what we see application teams develop – think what Google does! – all the Software-as-a-Service really has a strong focus on containers. So ideally, as an application team, all you want to focus on is you write your software, you test it, [and] you deploy it. And you do not have to focus on provisioning something – networking, storage, and all those things, or even virtual machines [VMs]. You just say: "Here's the software, deploy!" We have based SUSE Cloud Application Platform [7] on that; that is where Cloud Foundry [8] has a history and a lot of experience meeting user requirements. It's focusing on developing, testing, deploying, and not worrying about the infrastructure. That's why if you look at the SUSE architecture, at the top of the architecture, you have application delivery that's instantiated with SUSE Cloud Application Platform.

LM: Where does CaaS fit in?

GP: Where do you deploy stuff? And when I say stuff, increasingly often that stuff is actually containers. Some people package classic software into one huge container. For that, but also to be the foundation for something like Cloud Application Platform, a rich container platform like Kubernetes-based SUSE CaaS Platform is useful.

What if you could quickly and easily, without a lot of manual steps, deploy a cluster running a container infrastructure, have a nice dashboard where you see what's going on in the cluster, how your utilization is, where you're reaching some thresholds like storage or other things, where you can update the cluster, extend the cluster, and everything? For people on the infrastructure side, that's actually really important. In the past, you had one application on one machine, [and] then you had 10-12 of VMs running. And now on the same machine, you are running maybe 100 containers or more. So there is just a lot more of individual things to manage. And if you try to do that manually, that's a problem.

I've not met a CIO so far who told me they are getting more people. The equivalent of what we're doing for the developer with Cloud Application Platform – write your code, deploy, using automation, using tooling to really scale up and scale out – is what SUSE CaaS Platform does in the Kubernetes context. It gives more control. So I can easily envision, and that's what we're actually seeing, that those big lift and shift containers, where you take your old application and just put it in one container, you may deploy it directly on the Kubernetes cluster [and] interface it directly with CaaS or Kubernetes. And then where you have your Platform-as-a-Service, that's where you use something like SUSE Cloud Application Platform.

LM: Security seems to be one of the big issues in terms of deploying containers and managing them. Do you have any special groups that are focused on security?

GP: No SUSE product is ever going to leave the company without a security review.

LM: What does that entail?

GP: Our security team reviews the architecture of the product – for example, are things being encrypted when they go over the line? – and reviewing components that we ship for known security issues. Some of our security people actually are really good at breaking into systems. They are trying to break into our own products.

Security is rarely binary, because essentially everything is penetrable if you apply enough force. Sometimes the amount of force is not available to people, but if you are really determined, then maybe the weak link is not the security of the software; the weak link is the sys admin, and you put them under pressure. That's why I'm saying there is no point in using hundred-thousand-bit encryption apart from the fact that it will be terribly slow. You know, sometimes a wall is high enough, and building it higher doesn't actually help anything.

LM: Besides security, what are the other challenges of deploying open source software in the enterprise that you usually encounter when talking to people?

GP: I don't think I can answer that question for open source versus non-open source. For most people, open source is interesting; it's appealing. "If I have a choice, I want to avoid vendor lock-in" or "I like the principle." But if push comes to shove and you are the decision maker, you need the solution that is more secure, you need the solution that is faster, more performant, that uses less energy or space, and that has a lower cost of ownership.

By virtue of how we in the bigger community sense develop open source upstream and how we in the smaller SUSE sense productize it, we have demonstrated that in those areas where we decide to play, we actually will win. I mean, where is Unix, right? Which container solutions beyond Kubernetes can you think of? And I predict we'll see the same around Software Defined Storage where Ceph [9] is in a strong position to become the Linux of storage. ■■■

Info

[1] openSUSE: *https://www.opensuse.org*

[2] Dr. Gerald Pfeifer: *http://www.pfeifer.com/gerald/*

[3] Tumbleweed: *https://en.opensuse.org/Portal:Tumbleweed*

[4] openQA: *http://open.qa*

[5] Leap: *https://software.opensuse.org/distributions/leap*

[6] Closing the Leap gap: *https://lists.opensuse.org/opensuse-project/2020-04/msg00002.html*

[7] SUSE Cloud Application Platform: *https://www.youtube.com/watch?v=FBBTQL_swu8*

[8] Cloud Foundry: *https://susedefines.suse.com/definition/cloud-foundry*

[9] Ceph: *https://ceph.io/ceph-storage*

Monitor resource contention with Pressure Stall Information

Pressure Gauge

Pressure Stall Information (PSI) is a new feature that gives users a better view of resource contention. *By Paul Menzel*

C PU, RAM, and I/O are the three most important computing resources. If these resources are depleted, processes start to fight for them and resource contention scenarios occur. To avoid these problems, it makes sense to monitor precisely what resources applications use in order to correctly dimension the hardware requirements and make optimal use of the existing hardware.

Smartphones can also benefit from this type of monitoring: when resources become scarce, phones often terminate programs in the background in order to offer the foreground application sufficient resources.

Load Average

Most administrators keep an eye on the load average to determine the extent to which a system is exposed to load. The `uptime` or `top` (Listing 1) commands both display the load average, reading the values from the `/proc/loadavg` file. The file contains five values (Line 6). In addition to the three load average values, the fourth value counts the current executable kernel scheduling entities and the fifth shows the process ID of the latest process created.

The first three values state the load average for the executable processes, that is, the number of processes with a status of R (for runnable) that are waiting for CPU cycles and those with a status of D (for disk sleep) that are waiting for I/O. The three values show the load average for a period of 1, 5, and 15 minutes.

Few admins actually understand these values, and even the kernel developers don't actually think too much of them (Figure 1) [1].

Listing 1: Load Average

```
01 $ strace -e file uptime

02 [...]

03 openat(AT_FDCWD, "/proc/loadavg", O_RDONLY) = 4

04 16:17:55 up 3 days, 3:27, 1 user,  load average: 0,44, 0,78, 0,64

05 $ more /proc/loadavg

06 0.12 0.26 0.36 2/909 130726
```

```
    /*
     * kernel/sched/loadavg.c
     *
     * This file contains the magic bits required to compute the global loadavg
     * figure. Its a silly number but people think its important. We go through
     * great pains to make it work on big machines and tickless kernels.
     */
    #include "sched.h"
```

Figure 1: Comment by Linux scheduler maintainer Peter Zijlstra on `loadavg.c`.

The brief summary of their findings is: the load average is not useful when it comes to assessing system utilization. The admin can, at most, use the three values to estimate whether a reported performance problem has already been solved because the values become smaller again. However, computing the load average has other disadvantages:

- The admin has to interpret the values relative to the number of threads and CPUs. A value of 128 can be completely acceptable on a 128-thread system. On a 4-threaded system, however, probably nothing will work.
- The value does not tell you how long a process had to wait for resources.
- CPU and I/O are related, which makes it impossible to examine CPU and I/O requirements separately.
- Because the minimum resolution is one minute, the load cannot be observed in real time.

These problems make the load average unsuitable as a metric for admins who need to react quickly to performance events.

PSI

To better identify resource conflicts, and to make it easier to manage overloaded systems at Facebook's data centers, Facebook's kernel team developed Pressure Stall Information (PSI). According to the project's website [2], PSI is a canonical new way to obtain utilization metrics for memory, CPUs, and I/O via the Linux kernel.

Linus Torvalds published the 2018 implementation by Johannes Weiner in Linux 4.20. Weiner is also listed as the PSI maintainer [3]. In Debian 10 "Buster," this functionality is not yet included, but it can be found in Ubuntu from version 19.04 with a Linux kernel as of version 5.0.

Like other common kernel parameters, PSI values are accessed through the `/proc` pseudo-filesystem, which acts as an interface to kernel data structures. PSI values appear in three different files within the `/proc/pressure` directory:

- `cpu` – information related to CPU utilization
- `memory` – information on the time processes spend waiting due to memory issues
- `io` – information on the time spent waiting for I/O

Though the three files offer slightly different data, they are organized in a similar way.

CPU

The `/proc/pressure/cpu` file contains four values: `avg10`, `avg60`, `avg300`, and `total` (Listing 2, Line 2). The values starting with `avg*` represent the percentage of processes in the last 10, 60, and 300 seconds that had to wait for CPU resources.

Listing 2: Measuring with cpu

```
01 $ more /proc/pressure/cpu

02 some avg10=0.02 avg60=0.05 avg300=0.02 total=3990454056

03 [...]

04 $ watch -n ,1 grep -R . /proc/pressure/cpu

05 Every 0.1s: grep -R . /proc/pressure/cpu   machine: Mon Jun  8 17:54:34 2020

06 some avg10=0.00 avg60=0.00 avg300=1.07 total=2684388193
```

Suppose two processes each occupy 100 percent of one thread and run for five minutes on a two-threaded system where nothing else is running. In this case, all three averages are zero, of course, because the processes didn't have to wait. The fact that PSI delivers accuracy to within 10 seconds is an important benefit, because you can react 50 seconds earlier than with the load average.

The last value, `total`, also makes a big difference: the `total` value specifies the total time in microseconds for which processes had no CPU resources available. You can read this value at any time – about every 500 milliseconds. The difference between two successive values is the number of microseconds in this time interval in which processes had to wait for CPU resources.

This kind of granular resolution was not available previously in the Linux kernel. Thanks to `watch`, you can see how the value increases in realtime (Listing 2, Lines 4 to 6), which means you can monitor resource bottlenecks in real-
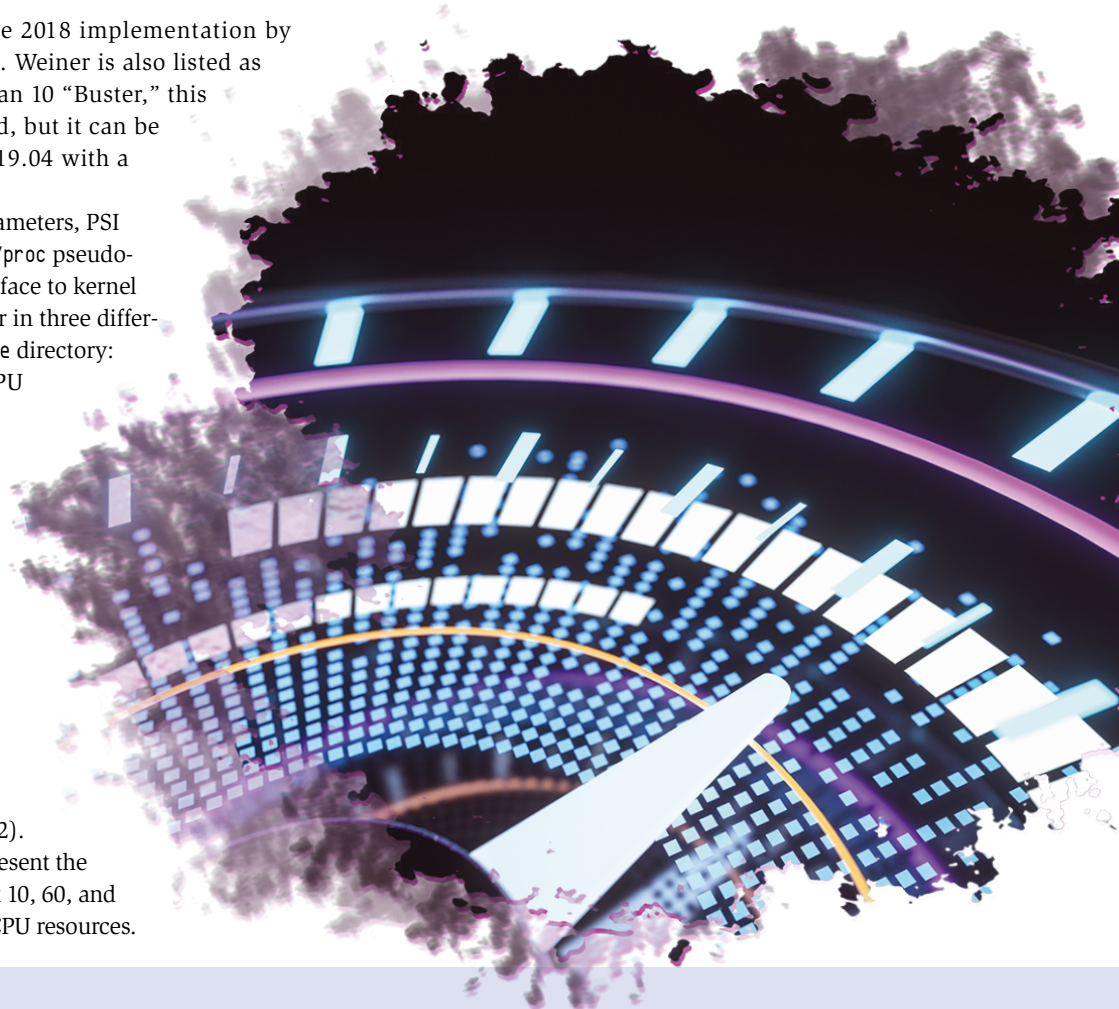
Listing 3: Measuring with memory and io

```
$ grep -R . /proc/pressure/

/proc/pressure/io:some avg10=0.00 avg60=0.00 avg300=0.00 total=10587199096

/proc/pressure/io:full avg10=0.00 avg60=0.00 avg300=0.00 total=10072568253

/proc/pressure/cpu:some avg10=30.27 avg60=29.97 avg300=18.80 total=1620253162

/proc/pressure/memory:some avg10=0.00 avg60=0.00 avg300=0.00 total=15411

/proc/pressure/memory:full avg10=0.00 avg60=0.00 avg300=0.00 total=12389

$ uptime

07:24:59 up 2 days, 16:15,  1 user,  load average: 150.58, 118.00, 76.42
```

Listing 4: Overloaded Backup Server

```
$ grep -R . /proc/pressure/

/proc/pressure/io:some avg10=15.60 avg60=11.13 avg300=7.98 total=94192093351

/proc/pressure/io:full avg10=15.60 avg60=11.13 avg300=7.97 total=93713900789

/proc/pressure/cpu:some avg10=0.00 avg60=0.00 avg300=0.00 total=1159442298

/proc/pressure/memory:some avg10=67.79 avg60=67.80 avg300=72.51 total=618948360599

/proc/pressure/memory:full avg10=67.60 avg60=67.58 avg300=72.18 total=613900281165
```

Listing 5: Polling Syntax

```
some|full Stall_Amount Time_Window
```

time and, if necessary, kill unimportant processes or migrate them to other systems.

Memory and I/O

The two other files, memory and io, each return two lines. The first line starts with some; the second with full. The some values show the portion of time in which at least one process is stalled, and the full values show the time in which all non-idle processes are stalled simultaneously. According to the documentation at the Kernel.org site, the full state means that "…actual CPU cycles are going to waste, and the workload that spends extended time in this state is considered to be thrashing." Listing 3 shows an example of a 2-socket compute node with an AMD EPYC 7551 and a total of 128 threads.

A large full value in memory can mean that the system was unable to handle a single runnable process in this time and that the CPU was probably busy paging. The overloaded backup server in Listing 4 illustrates

Listing 6: psi_example.c

```
01 #include <errno.h>
02 #include <fcntl.h>
03 #include <stdio.h>
04 #include <poll.h>
05 #include <string.h>
06 #include <unistd.h>
07 /*
08  * Monitor memory partial stall with 1s tracking
09  * window size and 150ms threshold.
10  */
11 int main() {
12   const char trig[] = "some 150000 1000000";
13   struct pollfd fds;
14   int n;
15   fds.fd = open("/proc/pressure/memory",
16                O_RDWR | O_NONBLOCK);
17   if (fds.fd < 0) {
18     printf("/proc/pressure/memory open error: %s\n",
19            strerror(errno));
20     return 1;
21   }
22   fds.events = POLLPRI;
23   if (write(fds.fd, trig, strlen(trig) + 1) < 0) {
24     printf("/proc/pressure/memory write error: %s\n",
25            strerror(errno));
26     return 1;
27   }
28   printf("waiting for events...\n");
29   while (1) {
30     n = poll(&fds, 1, -1);
31     if (n < 0) {
32       printf("poll error: %s\n", strerror(errno));
33       return 1;
34     }
35     if (fds.revents & POLLERR) {
36       printf("got POLLERR, event source is gone\n");
37       return 0;
38     }
39     if (fds.revents & POLLPRI) {
40       printf("event triggered!\n");
41     } else {
42       printf("unknown event received: 0x%x\n",
43              fds.revents);
44       return 1;
45     }
46   }
47   return 0;
48 }
```

this nicely. In this example, logging onto the system with SSH took more than a minute.

Polling

The Linux PSI interface lets admins generate triggers by writing them to the files and then reading them with poll(). Listing 5 breaks down the syntax; the values for the stall amount and the time window are in microseconds.

Listing 6 shows an example of a monitoring program from the Linux documentation [4]. The program defines an event that sends notifications if a process fails to receive RAM re-

sources for more than 150 milliseconds within a one-second time interval. If you name the file, say, psi_example.c, you can build it easily by typing make psi_example, assuming you have the build tools in place.

Conclusions

PSIs compressed to only one or two lines inform the admin about resource bottlenecks [5]. The file-based interface makes it easy to integrate scripts and helps to build monitoring systems. Even external system monitoring tools such as Atop already integrate PSI (Figure 2).

Thanks to the integration of PSI in Cgroups, admins receive this information globally for the entire system and in a granular form. PSI provides admins with a powerful alternative to the load average for a better overview of resource bottlenecks. ∎∎∎

Figure 2: Version 2.4.0 and newer versions of Atop also show the PSI.

Info

[1] "Solving the Mystery": *http://www.brendangregg.com/blog/ 2017-08-08/linux-load-averages.html*

[2] PSI: *https://facebookmicrosites.github.io/psi/*

[3] Weiner's project presentation: *https://lkml.org/lkml/2018/8/28/816*

[4] PSI documentation: *https://www.kernel.org/doc/html/latest/accounting/psi.html*

[5] Examples of using PSI: *https://unixism.net/2019/08/linux- pressure-stall-information-psi-by-example/*

Easy steps for optimizing shell scripts

Speedier Scripts

Shell scripts are often written for simplicity rather than efficiency. A closer look at the processes can lead to easy optimization. *By Jörg Schorn*

A shell script is essentially a sequence of commands and branches. Listing 1 shows a typical example. `query.sh` has all the basic components of a shell script. It contains keywords such as `if`, `then`, `else`, and `fi`; shell built-in commands such as `echo` and `read`; the square bracket; and last but not least, an external command (`/bin/echo`).

The calls to `echo` and `/bin/echo` behave differently, although they give you the same results. To see the difference, call the script twice while monitoring the shell with the command `strace`.

Listing 2 shows the first pass; Listing 3 uses Strace to display the corresponding output. At the start, the system creates a new shell process with a process identifier (PID) of 2489, which is responsible for processing the script. If you say `y` at the prompt, the output appears and the script is terminated (Listing 3, last line).

Now call the script a second time and say `n` when prompted (Listing 4). The branch now causes a call to the external command `/bin/echo`. Strace shows completely different output (Listing 5). In this case, a new shell process also started at the beginning (PID 2510). However, another process with a PID of 2511 is also created: This is the PID of `/bin/echo`.

Listing 1: query.sh

```
#!/bin/bash
echo "Own process ID: $$"
echo -n "Are you an admin? "
read answer
if [ "$answer" = "Y" -o "$answer" = "y" ]; then
  echo "You are an admin."
else
  /bin/echo "You are not an admin."
fi
```

Listing 2: First Pass

```
# ./query.sh
Own process ID: 2489
Are you an admin? y
You are an admin.
```

Listing 4: No This Time

```
# ./query.sh
Own process ID: 2510
Are you an admin? n
You are not an admin
```

Listing 3: Strace for echo

```
clone(child_stack=NULL, flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLD,child_tidptr=0x7fa8fc515e50) = 2489

strace: Process 2489 attached

[pid  2489] +++ exited with 0 +++

--- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_EXITED, si_pid=2489, si_uid=0,si_status=0, si_utime=0, si_stime=0} ---
```

When a script calls external commands, the shell creates a new process, whereas the use of the commands built into the shell, such as echo, does not require an additional process. Figure 1 shows a schematic representation of the two examples. The graphic does not claim to be complete; it is only intended to show how the two call variants behave.

Subprocesses

The shell acts as an interactive interface between the user and the operating system. When you call a command, the shell stores the return values in environment variables. Most shell commands are small programs of their own. Each program requires its own address space, registers, and other resources. Unix-like operating systems use a system call to fork() (on Linux this is clone()) to create a copy of the calling process. Within this copy, you then start the actual target program with the exec*() system call, which replaces the copied process.

The shell creates at least one additional subprocess when calling an external command, substituting commands, or using anonymous pipes. Using a function integrated into the shell, however, does not create a new subprocess.

Anonymous pipes are created by the | sign within a chain of commands. They are referred to as anonymous because they are not normally visible to the user – the system creates them and deletes them again after processing the data.

A process on Unix usually has three channels: STDIN (channel 0), STDOUT (channel 1), and STDERR (channel 2). The pipe normally reads the data from STDIN and outputs the data to STDOUT. Error messages are sent to STDERR.

If you are working with anonymous pipes, the first process redirects its output to STDIN of the process that follows the pipe symbol instead of STDOUT. In Listing 6, for example, the output from ls provides the input for the wc command. The output from wc is then ultimately sent to the standard output.

You can only read from a pipe while a process is writing data to it. This explains why an anonymous pipe creates subpro-

Listing 5: Strace with /bin/echo

```
clone(child_stack=NULL, flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLD,child_tidptr=0x7fa8fc515e50) = 2510

strace: Process 25attached

[pid  2510] clone(child_stack=NULL, flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLD, child_tidptr=0x7fbce6ffee50) = 2511

strace: Process 2511 attached

[pid  2511] +++ exited with 0 +++

[pid  2510] --- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_EXITED, si_pid=2511,si_uid=0, si_status=0, si_utime=0, si_stime=0} ---

[pid  2510] +++ exited with 0 +++

--- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_EXITED, si_pid=2510, si_uid=0,si_status=0, si_utime=0, si_stime=0} ---
```
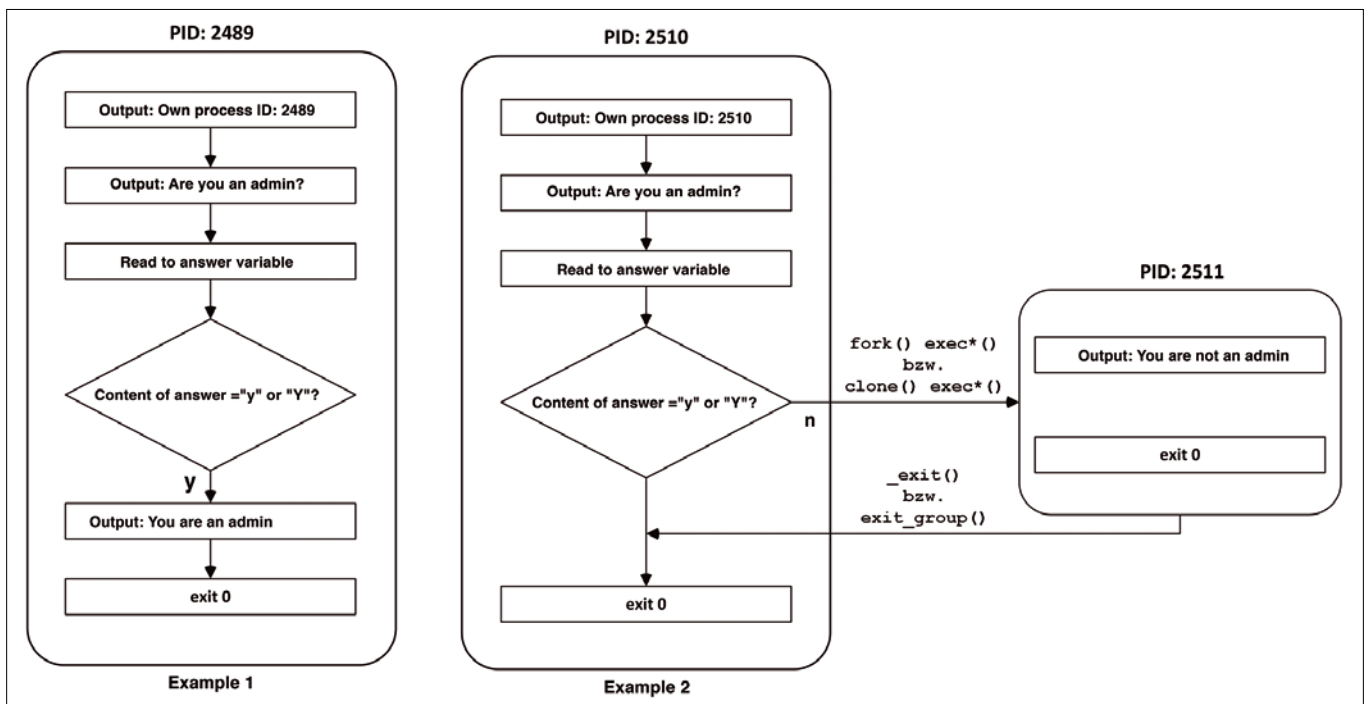

Figure 1: On the left, you can see the call to the shell's built-in echo command. The output on the right is created using the external /bin/echo command.

Listing 6: Pipe

```
$ ls -l | wc -l

282
```

Listing 7: Command Substitution

```
# filecount=$(ls | wc -l)

# echo $filecount

28
```

cesses: The `wc` command reads from the pipe as long as the `ls` command writes to it.

Anonymous pipes are also used for command substitution (Listing 7). In this case, the shell opens an additional channel that is used to read data – channel 3, if possible. The output of the concatenated command ends up in the `filecount` variable.

Script Tuning

Probably the most important rule for fast scripts is to avoid unnecessary subprocesses. Following are some examples to illustrate the effects of excessively long pipelines and command substitutions. The setup for this test includes 100,000 files in a folder, some of which contain the string `50` in one line.

In the first example (Listing 8), a loop evaluates each file. To do this, the script outputs each file with the `cat` command and redirects the output to the `grep` command. `grep` looks for lines containing the string `50` – so the string must be exactly two characters long, start with a `5`, and end with a `0`.

The output from `grep` again provides the input for the `wc` command, which counts the lines found. The output from `wc` ends up in the `found` variable. If this variable assumes a value of 1, the script outputs the file name. Table 1 shows the results for this test and three other variants, each determined by the `time` command.

Listing 9 shows roughly the same approach, but this script does not store the output in a variable and does not determine the number of lines per file consisting of exactly the string `50`. Because it requires fewer subprocesses, the modified script takes about 60 percent of the original execution time (see Table 1).

In the next step, the number of processes is reduced again, because the `grep` command itself opens the file and searches for the desired line. Then the script queries the return value and prints the file name if a matching line exists (Listing 10). The script now needs only about 30 percent of the original time to examine all files.

However, you can perform this task even faster. The `grep` command has a parameter that it uses to output the file name in case of a hit, instead of all lines

Listing 8: cat and grep

```
#!/bin/bash
# Example 1
for file in *; do
   found=$(cat $file|grep "^50$"|wc -l)
   if [ $found -ge 1 ]; then
     echo $file
   fi
done >number_files_1.out
```

Listing 9: A Little Faster

```
#!/bin/bash
# Example 2
for file in *; do
   cat $file|grep -q "^50$"
   if [ $? -eq 0 ]; then
     echo $file
   fi
done >number_files_2.out
```

Listing 10: Even Faster

```
#!/bin/bash
# Example 3
for file in *; do
 grep -q "^50$" $file
 if [ $? -eq 0 ]; then
   echo $file
 fi
done >number_files_3.out
```
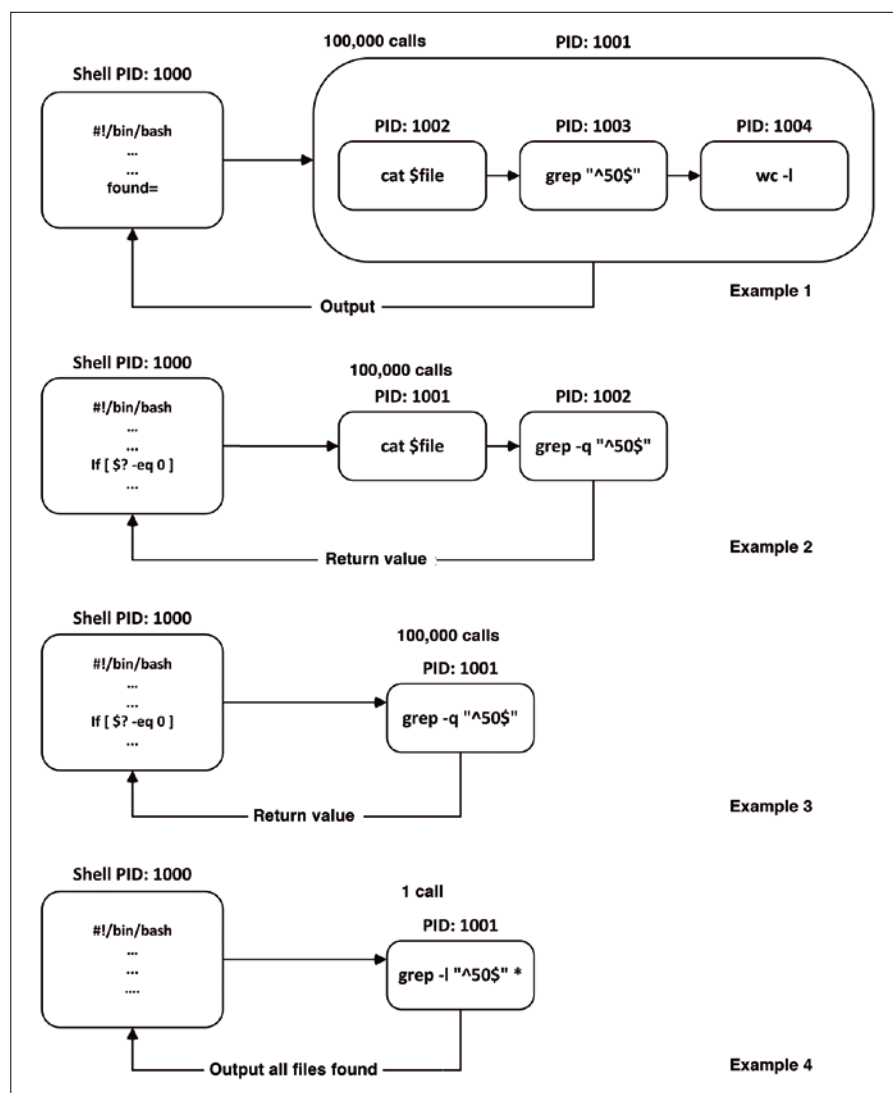

Figure 2: Due to the different command chains, the variants require a different number of subprocesses for the same action.

Table 1: Grep Timer

Category	Listing 8	Listing 9	Listing 10	Input
real	3m52.327s	2m36.079s	1m2.970s	0m0.372s
user	3m11.611s	1m40.114s	0m46.838s	0m0.083s
sys	0m30.046s	0m49.343s	0m13.408s	0m0.264s

Listing 11: Fastest

```
# time grep -l "^50$" * >number_files_4.out
```

of a file containing the string to be searched. A single call to the command is sufficient to check all files and output the names of the files (Listing 11).

The execution time is reduced to one six-hundredth of the original script from Listing 8. The enormous increase in speed is due to the fact that the shell now only creates one sub-process for the grep command. Figure 2 roughly sketches how the individual variants of the scripts differ.

In the first example, running through 100,000 files requires four subprocesses each, for a total of 400,000. In the second example, two subprocesses per file are sufficient, for a total of 200,000. In the third example, this value is halved again, resulting in only 100,000 subprocesses. At the command line, however, a single additional process does the job.

Monitor the scripts at run time with Strace, and log the system calls that create new processes. A check using Grep for

the clone keyword confirms the above values. To make sure that all variants return the same results, compare the output of the first example with all other output if necessary – there should be no difference.

Leveraging the Options

Many roads lead to Rome, but there are ways that will get you to your desired destination faster. In order to tweak the most speed out of scripts, you should give the commands you use as many tasks as possible.

For example: assume you want to print every third line from a file. The simplest way is to use tail to print all the lines starting at the desired position and redirect this output to the head command, which in turn only prints the first line from the lines it parses.

It is obvious that this solution is slow. It makes sense to move straight on to a more promising option: a loop with an incremented run index (Listing 12). If the index has a residual value of 0 when divided by 3 (modulo), the script outputs this line using the sed command. For a text file with 10,000 lines, the search and output took about 3.5 seconds (see Table 2).

A far faster approach is to use Awk to read the file and determine every third line within the tool. The internal NR variable, which the software increments for each imported data record (usually one line), is fine for this purpose (Listing 13). Here the execution speed is many times higher, since the entire evaluation takes place within a single command.

Table 2: sed Timer

Category	Listing 12	Listing 13
real	0m3.474s	0m0.026s
user	0m2.591s	0m0.006s
sys	0m0.438s	0m0.004s

Listing 12: sed at Work

```
#!/bin/bash

typeset -i i=1

while [[ $i -le 10000 ]]; do

 if (($i % 3 == 0)); then

    # Runtime index modulo 3

    sed -n "$i p" textfile

 fi

  ((i++))

done
```

Listing 13: Awk Instead

```
#!/bin/bash

awk '{

   if ((NR % 3 == 0))

     # Index modulo 3

     print $0

}' textfile
```

Computation

You have several different options for calculating the sum of numbers that a text file contains. In the example in Listing 14, every line containing the `fiftieth` string is interesting. The script evaluates a file that contains one million lines. Every `fiftieth` line contains the string.

Here, too, you can use Awk as a tool for quick summation (Listing 15). But Awk does not work directly in machine language. For this reason, it makes sense to pass the search for the character string to the `grep` command and then add the lines found using Awk (Listing 16).

In this example, too, optimization achieved significant speed

Listing 14: Looking for a String

```
#!/bin/bash

typeset -i sum=0

while read line; do

  set $line

  # Totaling field 6

  sum=$((sum+$6))

# Parse output from grep

done < <(grep " fiftieth " largefile)

echo "Sum total: $sum"
```

Table 3: Awk Timer

Category	Listing 14	Listing 15	Listing 16
real	0m4.471s	0m1.408s	0m0.231s
user	0m2.374s	0m1.348s	0m0.050s
sys	0m1.956s	0m0.013s	0m0.010s

gains. The variant from Listing 15 reduces the runtime to about one third; the variant from Listing 16 runs twenty times faster than the first alternative (see Table 3).

Conclusions

The examples in this article show that you can drastically increase the speed of your scripts by skillfully using multifunctional tools such as Awk, Python, or Perl and by avoiding complex constructs with Tr, Sed, or Grep: You will thus consistently avoid many context changes.

However, not every anonymous pipe is detrimental to throughput, as the last example shows. Instead, it is more important to use the strengths of the various tools and keep the number of subprocesses as low as possible.

It is also important to remember that you can improve the readability and thus the maintainability of the scripts by doing without complex chains of commands. ■■■

Listing 15: Faster with Awk

```
# time awk '/ fiftieth / {sum += $6} END {print "Sum total:", sum}' largefile
```

Listing 16: Fastest: Awk with grep

```
# time awk '{sum += $6} END {print "Sum total:", sum}' < <(grep " fiftieth " largefile)
```

Tips for speeding up your Linux system

Tweak Talk

If you are looking for ways to speed up your Linux, consider this collection of curated performance tweaks. *By Alexander Tolstoy*

L inux is renowned as a high-performance operating system, and it runs on nearly all of the world's most powerful supercomputers. It also runs very well on regular desktops and workstations, but sometimes people ask for more. Whether you're faced with a low-end hardware setup or a loaded production system with high I/O, there is always room for tweaks and optimizations. Linux is an ideal OS for tinkering, and you have many options for eliminating performance bottlenecks, fixing non-optimal settings, and making the system more fluid and responsive. The goal of this article is to point you to some best practices for tweaking a typical home or office Linux-powered machine, while avoiding some of the outdated or less efficient advice.

The Curse of Low Memory

"Buy more RAM" – that's a frequent response to "I've got only 2GBs." However, sometimes it is not possible to install more memory bars into a computer. An average Linux desktop runs butter-smooth with 8GB, very nicely with 4GB (with some limitations to multitasking), and quite poorly with 2GB or less. Some palliative techniques that bring relief include using zram and zswap. These are the two methods of compressing memory to take down (or even completely avoid) swapping memory pages to the hard drive. Thanks to compression, the system has more free RAM, and with the lower swapping, the filesystem also speeds up. The trade-off is a higher CPU load due to constant compressing and decompressing, but its impact is usually smaller than the lagging caused by a running out RAM.

Zram is a compressed RAM-based swap device designed for systems with no physical swap partitions. It is a Linux kernel module (included since kernel 3.14) that creates a very fast virtual block device backed by RAM and sets it as a top-priority swap "partition." All you need to do is install the supplementary package for the zram systemd service and enable it. In Ubuntu, use the following commands:

```
$ sudo apt install zram-config
$ sudo systemctl start zram-config
```

Now there are extra `/dev/zramX` virtual devices, one per each of your CPU cores. It is easy to see the new swap setup with the `$ sudo zramctl` command (Figure 1). The size of the devices is determined automatically based on your system's specs, although you can still change it. The following example sets `/dev/zram0` to 1GB:

```
# swapoff /dev/zram0 && zramctl ⤶
--size 1024M /dev/zram0
# mkswap /dev/zram0 && swapon /dev/zram0
```

Zswap is different in that it is designed for systems with a normal swap partition. If there is a swap partition, zswap provides a

Author

Alexander Tolstoy is a long-time Linux enthusiast and tech journalist. He never stops exploring hot new open source picks and loves writing reviews, tutorials, and various tips and tricks. Sometimes he must face a bitter truth thanks to the inhuman `fortune | cowsay` command that he thoughtlessly put in `~/.bashrc`.

Figure 1: Once the zram service for systemd is started, extra compressed swap devices appear.

compressed cache that sits between RAM and the swap device. When it is time to offload memory pages to disk, zswap compresses the pages and stores them in the dedicated RAM cache. This technique is often referred to as a *write-back cache*, and it significantly reduces the disk I/O. Zswap is a kernel feature (version 3.15 onwards) that you can enable and configure via the GRUB2 bootloader parameters – in the most basic case, it is enough to add `zswap.enabled=1`. I'll explicitly define the compression algorithm as LZ4, tell zswap to allocate 30 percent of RAM, and set the allocator to z3fold for better memory management:

```
zswap.enabled=1 zswap.compressor=lz4 zswap.max_pool_percent=30 ⇗
zswap.zpool=z3fold
```

It is theoretically possible to use both zram and zswap at the same time, but this option is not known to improve performance more than the improvement you get from choosing one of the tools. When deciding on a compressing method for your system, keep in mind that zram works best in situations where you want to avoid HDD or SSD wear out (e.g., on small systems with flash storage). Zswap is a more general-purpose tweak.

More on Swapping

Depending on the amount of available RAM, it may be worth adjusting parameters such as swappiness and disk cache pressure. Both take settings from `/etc/sysctl.d/99-sysctl.conf`. There is a popular mistaken belief that swappiness represents a threshold for RAM usage, and when the amount of used RAM hits that threshold, swapping starts. In fact, the `vm.swappiness` setting is a ratio (0-100) that controls the aggressiveness of swapping, particularly the memory pages reclamation by the kernel. Changing swappiness only makes sense for systems with slow or aging mechanical drives in order to reduce the swapping activity to some degree. For instance, changing the value from 60 (default) to 40 is a good practice for making a slow machine perform better, especially if it has at least 2GB of RAM.

Disk cache pressure (`vm.vfs_cache_pressure`) controls the tendency of the kernel to reclaim the memory that is used for caching of directory and inode objects. Setting the value to 0 leads to out-of-memory lockdowns, whereas increasing it over the default value of 100 makes the Linux kernel reclaim VFS inodes too actively and slows down the system. The optimal value is perhaps 50; at least my test showed that setting the value to 50 makes many filesystem operations (like finding files) noticeably faster.

Tinkering with the Kernel?

There are lots of third-party patches for Linux that require manually applying and rebuilding the whole source tree of the

kernel, which could be a difficult and time-consuming task. The promising, efficient, and easy way to include these options is the XanMod kernel.

The XanMod project provides a patch set and prebuilt kernel packages for Debian, Ubuntu, Arch, and Gentoo. This modified kernel features the optimized CPU scheduler, Budget Fair Queueing (BFQ) I/O scheduler (the tool that controls the disk read/write requests management, which plays a huge role in keeping the OS responsive when the disk is under heavy load), and other tweaks for better preemptive multitasking and lower latencies. When compared to the stock Linux kernel in such tests as encoding something with FFmpeg, resizing and rotating images in Gimp, and similar CPU-heavy tasks, the XanMod 5.4 kernel was 8-10 percent faster, which was a solid margin. Install XanMod in Ubuntu as follows:

```
$ echo 'deb http://deb.xanmod.org releases main' | sudo tee ⇗
/etc/apt/sources.list.d/xanmod-kernel.list && wget -qO ⇗
- https://dl.xanmod.org/gpg.key | sudo apt-key add

$ sudo apt update && sudo apt install linux-xanmod
```

Hard Disk Requests Scheduler

Many of XanMod's benefits depend on the I/O scheduler. Historically, Linux has supported many I/O schedulers, but these days, the choice comes down to Multiqueue Deadline (`mq-deadline`), Kyber, and BFQ. Check their availability with this command:

```
$ cat /sys/block/sda/queue/scheduler
```

Multiqueue Deadline is an adaptation of the original Deadline scheduler, which was created to guarantee a start service time for a request. Our tests showed that `mq-deadline` is a good all-rounder with no drawbacks, yet with few advantages as well.

Kyber is a more recent scheduler tuned for fast multi-queue devices, such as modern NVMe drives. It has two queues: one synchronous and another for asynchronous requests.

BFQ is often advertised as the best scheduler of all. In fact, it provides the best interactivity for systems with relatively slow drives, including the low-end SSDs. If you wish to eliminate the slow I/O bottleneck and make disk read/writes appear to be faster, BFQ is second to none. Switch to the desired scheduler in the runtime using the following template:

```
$ sudo echo scheduler_name > /sys/block/sda/queue/scheduler
```

In the case of BFQ, it also makes sense to add the `scsi_mod.use_blk_mq=1` boot option in the GRUB configuration (Figure 2).

Changing the scheduler affects the Linux system when it is under heavy workload. For instance, try to encode or archive something big (or otherwise load the CPU), and then try to copy some data on a flash thumb drive. Such an exercise will show how the system handles the huge flow of I/O requests, which could help you decide with the right scheduler.

Make the Processes Run Nicer

Modern Linux systems include several services that start during boot time. Some of these services may be unused and

therefore disabled with no harm. The first step is to look at which services are consuming the boot time:

```
$ system-analyze blame
```

This command prints the list of auto-started services in the descending order based on how much time they took to start. If you don't know what a service is used for, don't disable it. However, if you are not managing a mail server, then it is safe to turn off Postfix. Also, consider if CUPS (for printing) and database services like PostgreSQL or MariaDB are there for any use. Disabling a service is as simple as:

```
$ sudo systemctl disable service_name
```

Use `stop` instead of `disable` to stop a service immediately. For the rest, re-sort and optimize the running processes using Ananicy. Ananicy [1], a third-party tool, consists of a shell script and a systemd daemon to control priorities of running processes and applications. This is solely a desktop-oriented tweak intended to solve such issues as, "Why does my game lag during kernel compilation?" Ananicy ships with a community-maintained list of rules, which are very sane for the most part. After installing the tool (follow the `README.md` guide), enable and start Ananicy as follows:

```
$ sudo systemctl enable ananicy && sudo systemctl ⤷
  start ananicy
```

The effect of tools such as Ananicy will be different across the endless variety of configurations, but it is always noticeable. Ananicy is perfect for laptops – it makes batteries last longer and fans behave more quietly.

Apply Filesystem Tweaks

In the past, it was common to hear the advice of using the `no-atime` mounting option for partitions listed in `/etc/fstab`. This option is counterproductive in modern Linux systems that already use the less risky and performance-friendly `relatime` option by default.

For everyday scenarios with most major Linux systems, the most balanced and fastest filesystem is usually ext4, which is already tuned for the best performance in most mainstream distros. However, long-running Linux systems suffer from disk fragmentation, which leads to slowdowns. Fragmentation in Linux isn't a nightmare that it used to be in the Windows world in late 90s, but it is still a problem if your Linux machine runs intensive disk reads and writes for a long time. The solution is defragmenting. There is the universal script by Con Kolivas, `defrag` [2], that rewrites files in order of largest to smallest and works for any filesystem. But, for the extX filesystem family, a better solution is shipped

within the *E2fsprogs* package. Start by examining the current state of fragmentation for a test `/dev/sda3` partition:

```
$ sudo e4defrag -c /dev/sda3
```

Look at the 'Fragmentation score' and see if it is not too high (e.g., below 30). Even if the partition is healthy, the above command will list the files that have been fragmented and may suffer from longer access times. It is easy to defragment them with this command:

```
$ sudo e4defrag /dev/sda3
```

The `e4defrag` command also accepts directories, so that you don't have to process the whole partition if you only work with a given directory. More than that, defragmenting adjusts free extents to the size of the files that you store. So, if you run the `e2freefrag` tool afterwards, you'll see the table of extents of different sizes, adapted to the kind of information that already exists on the disk (Figure 3):

```
$ sudo e2freefrag /dev/sda3
```

In addition to traditional defragmentation, you can also improve ext4 filesystem performance by reallocating frequently used files using e4rat. This tool reduces disk access time utilizing the `EXT4_IOC_MOVE_EXT ioctl` feature of ext4 and doing so-called *online defragmentation*. Modern e4rat code [3] takes few minutes to build from source (see the project page for guidance). Normally, e4rat requires three phases: first for learning (collecting files), second for reallocating what has been collected, and third for preloading reallocated data to page cache (Figure 4). To get started with e4rat, enter:

```
$ sudo systemctl stop auditd // (auditd conflicts with E4rat)
$ sudo e4rat-collect // (start opening apps you want to ⤷
  optimize, hit Ctrl+C when done)

$ sudo e4rat-realloc e4rat-collect.log
```
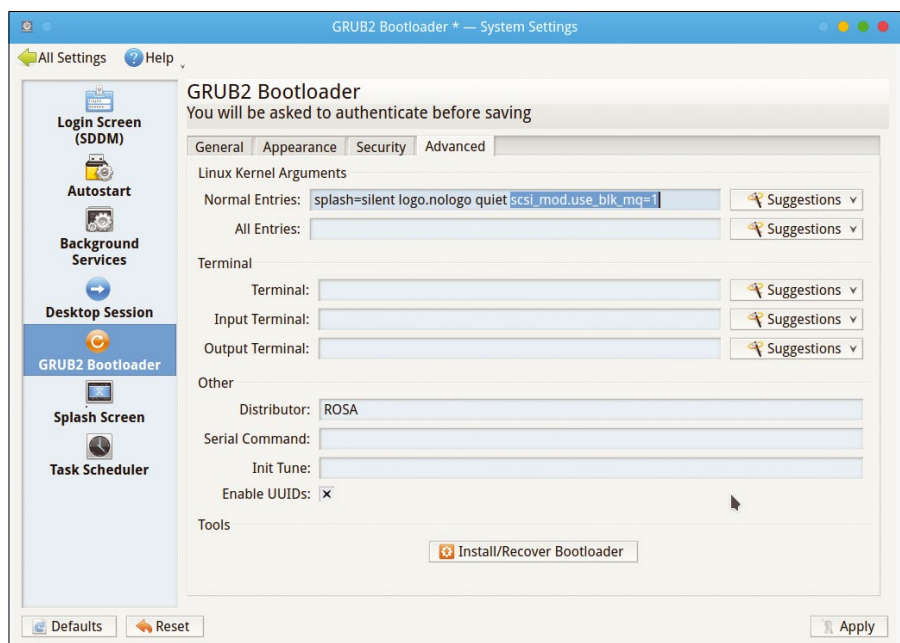

Figure 2: To make changes persistent, edit the GRUB2 configuration.

This technique effectively makes every cold start of an optimized app feel like it is hot, which is a great aid for slow or low-end Linux systems.

Small Tweaks

Also consider the following small, yet efficient tweaks:

• *Eliminate excessive cryptographic routines that involve hard drives.* To prevent the I/O on your SSD or HDD from contributing to the entropy pool, you can disable the `add_random` setting for your block devices:

```
# echo 0 > 🠖
    /sys/block/sda/queue/add_random
```

Linux uses the entropy pool for things like generating SSH keys, SSL certificates, or anything else crypto. Preventing your SSD from contributing to this pool probably isn't a security issue, but it will save you small amounts of I/O.

• *Bring parallel drive probing back.* It may seem surprising, but modern-day Linux distributions still probe ATA devices serially by default, which is called staggered spin-ups (SSS). This technique goes back to a time when spinning up multiple drives at once caused power usage peaks, and thus the Linux kernel avoided parallel probes. This behavior makes little or no sense on modern systems, especially if there are no rotational drives. Enabling probing ATA drives in parallel speeds up the boot process. First, check if the `SSS` flag is set in your system:

```
$ dmesg | grep SSS
```

If it is, add the `libahci.ignore_sss=1` boot option to GRUB. A Linux system with several hard drives or SSDs will see a better boot time.

• *Mount /tmp to RAM.* Most Linux systems already use `tmpfs` (check it with `$ df`), but you can add another RAM-backed mount point to get rid of temporary runtime clutter that usually sits in `/tmp`:

```
$ echo "tmpfs /tmp tmpfs rw, 🠖
    nosuid,nodev 0 0" | 🠖
    sudo tee -a /etc/fstab
```

This way your system will clear everything found in `/tmp` upon every reboot. Doing the same for `/var/tmp` is not recommended as long as the `/var/tmp` directory is meant to store data persistently.
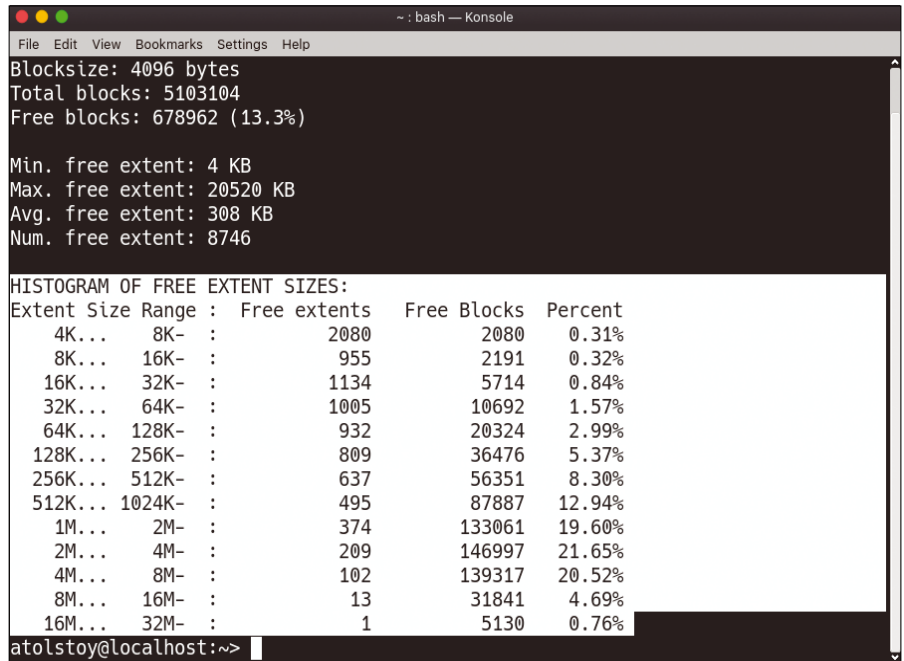
Figure 3: Too many small extents will lead to file fragmentation on an ext4 filesystem.

Conclusion

The very versatile Linux includes many modules, tools, and settings that allow you to tweak the system and improve performance. If you have a computer with insufficient memory or slower-than-expect I/O, or even if you just want to experiment to learn more about Linux, the techniques described in this article will help you take your first steps. ∎∎∎

Info

[1] Ananicy: *https://github.com/Nefelim4ag/Ananicy*

[2] defrag: *ck.kolivas.org/apps/defrag/defrag-0.08/defrag*

[3] e4rat: *https://github.com/leandromqrs/e4head*
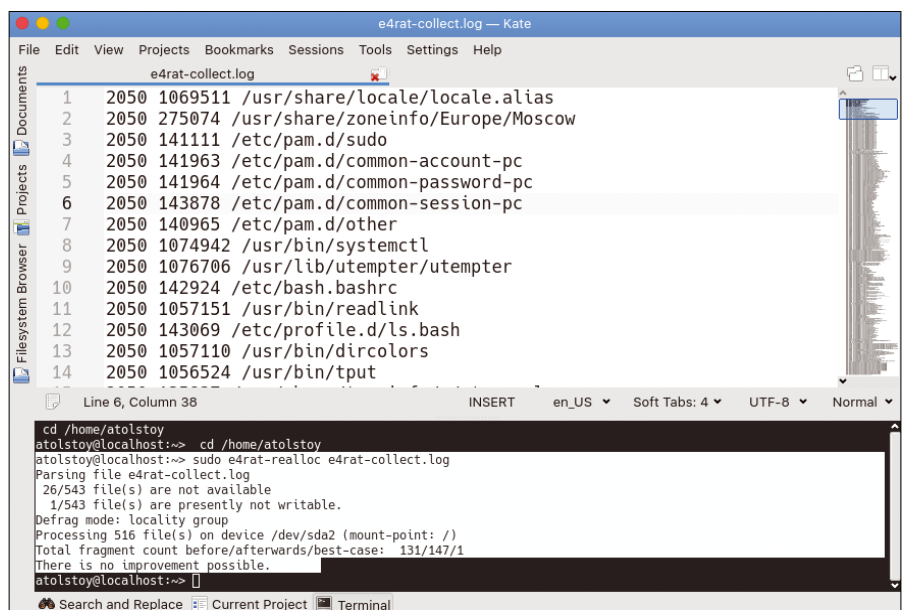
Figure 4: Reallocating frequently used data can be a magic potion for systems with rotational disks.

Distros, documentation, and support

Support System

Contrary to popular belief, free software does offer support. Depending on the distro, this support can be found in a wide range of documentation formats. Here's what to expect from a sampling of distros.

By Bruce Byfield

A common criticism of free software is that it comes with no support. That is not entirely true, since most projects maintain man pages and, in the case of the GNU Project, info pages. However, man and info pages are mostly for developers. To compensate for the lack of more general documentation, many projects, including distributions, have resorted to a wide variety of media for support, including blogs, online manuals, FAQs, how-tos, forums, and IRC channels.

These methods of documentation can be an important factor in selecting a distribution, especially for beginners. Although all sorts of documentation are available online, a distro's docs are a handy place to start. This allows users to avoid sorting through the glut of information on the Internet, much of which may be obsolete. Sometimes, other distros can also be a source of useful information, although much depends on how that information is maintained and organized.

Which distributions provide the most useful information? What follows is a sampling of distros' documentation (including Bodhi Linux and openSUSE, both of which you'll find on this month's DVD).

Arch Linux

Arch Linux tends to go its own way in installation, package management, and other features. Perhaps that is why Arch has always emphasized documentation. In fact, Arch documentation goes far beyond the distribution itself and also covers Linux in general. If your own distribution fails to explain a topic, you may very well find useful information on the Arch website.

Arch Linux's homepage is mostly for developers, listing changes in packages. By contrast, documentation is listed on ArchWiki [1] (Figure 1). For Arch Linux itself, the wiki links to a comparison of Arch with other Linux distributions and an installation guide, both of which are essential reading for those considering installation. In addition, under *General recommendations*, the wiki links to tutorials and a list of packages. Scroll down to find links to general information, such as the project's code of conduct and the best way to use the wiki.

Arch's user forums [2] contain an Arch-centric section. Otherwise, most of the forums concern general Linux issues, making the forums a place to start searching if you are using another distribution.

Bodhi Linux

Bodhi maintains a wiki [3] with general information to get users up and running, including a guide to Moksha, its graphical interface, and some links about how to customize an installation. The first time you log into Bodhi, the landing page opens with a quick start that summarizes basic tasks (Figure 2). Its forums [4] are small, and their traffic is usually limited to several messages a day. For more advanced help, users should go to Ubuntu, from which Bodhi is derived (see below).

Debian

Debian probably offers the most documentation of any distribution. That is good news, in that you can probably find information on any topic. The bad news is that you have to sort through a massive amount of loosely organized information, some of which is obsolete. It often seems that, once a link is posted, no one ever reviews it.

The Debian website has separate pages for developers [5] and packages [6]. However, most documentation resides on yet another page [7]. The documentation page starts with links to an installation guide and FAQ and then suggests printing out a reference card if you are new to Debian. A massive number of
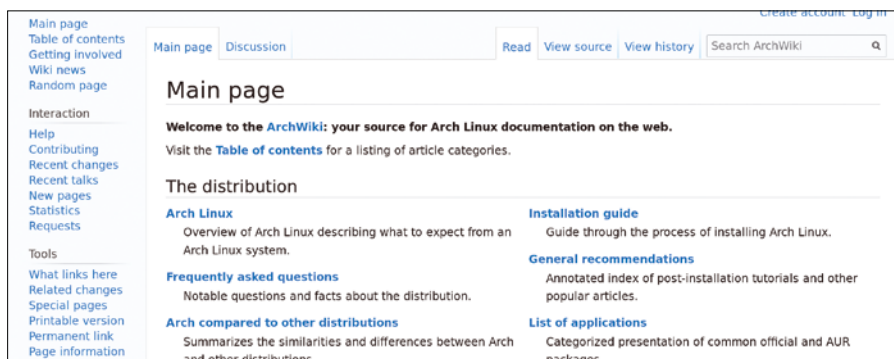
Figure 1: Arch Linux's wiki includes information that users of other distros often find useful.
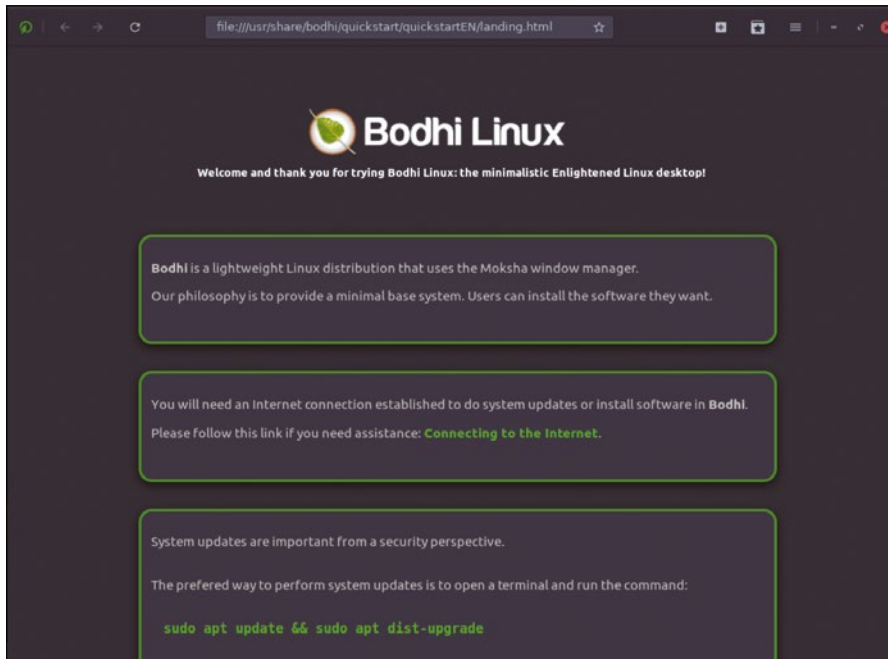
Figure 2: Bodhi includes an online quick start.
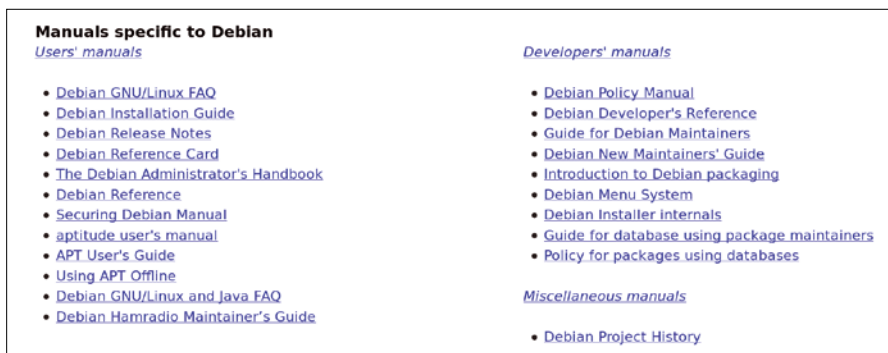
Figure 3: Debian documentation is extensive, including numerous manuals.

manuals are available: 12 for users and 9 for developers (Figure 3). A support page goes on to list other places to get help, such as IRC channels, Usenet groups, and the bug tracking system. There is even a consultants page, although there is no guarantee of the quality of support.

Debian's user and development forums are equally massive [8], both in number and in traffic. General users might want to begin with the debian-user list, which has especially heavy traffic, and usually generates responses within 20 minutes.

Fedora

Fedora has one of the most organized systems of documentation. Unfortunately, though, it is centered entirely on Fedora, making it less relevant to users of other distros. Its User Documentation page [9] covers various flavors of Fedora. The first two are the current and previous releases

of Fedora (currently 32 and 31). Below them are links to Fedora IOT, SilverBlue, and CoreOS. Still further down are links to draft documentation for Rawhide, Fedora's development repository, and links for obsolete documentation. In contrast to the semi-chaos of Debian documentation, this arrangement makes for efficient searches.

At the bottom of the documentation page is information about the project it-

self. Information about the Fedora leadership, including the council and engineering teams, is given, as well the project's budget and its involvement in the Outreachy and Google Summer of Code mentoring programs. Several other distributions display similar information, but separately from their technical documentation, which means it is often harder to find than on Fedora.

You'll find forums listed on the Ask Fedora page [10]. Although English is the dominant language for the forums, five other languages are also supported. Sources outside the project on sites like Reddit and Facebook are also listed. For non-Fedora users, *Fedora Magazine* [11] (Figure 4) is often the most useful source.

Linux Mint

Since Linux Mint is derived from Ubuntu and Debian, the documentation from these distributions can often help Linux Mint users. However, Mint maintains its own user, troubleshooting, and installation guides in two dozen languages [12]. The installation guide is particularly well-written, being divided into tasks. A developer's guide to the Cinnamon desktop environment is also available [13] (Figure 5). The main source of help for Mint is its high traffic user forums and IRC channels [14].

openSUSE

OpenSUSE places a high priority on documentation [15], although mostly for itself. It includes a number of guides in HTML, PDF, and EPUB, but, increasingly, the emphasis is on shorter, task-oriented documents [16] – including 125 how-tos. General information about the project includes an FAQ and guides to Zypper, YaST, and other tools specific to openSUSE. General information about Linux explains key con-
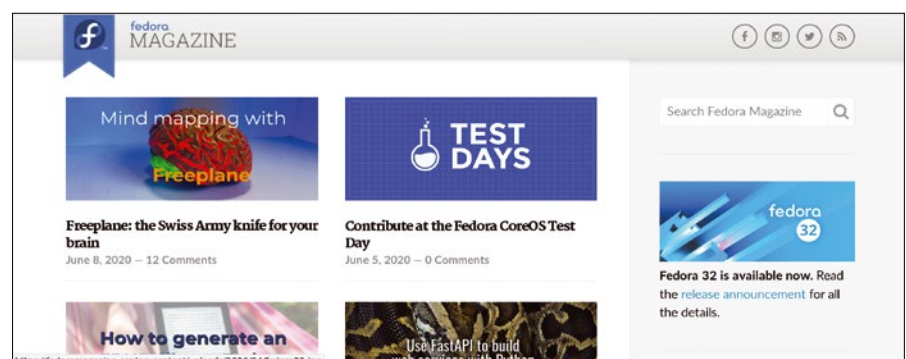
Figure 4: *Fedora Magazine* is a source for both Fedora and non-Fedora users.

Figure 5: In addition to basic user help, Linux Mint includes documentation for its popular Cinnamon desktop.

cepts and acronyms and gives a few hints for migrating from Windows.

Increasingly, openSUSE's main emphasis appears to be on online support [17] (Figure 6). The Support portal divides online help into non-interactive, interactive, and bug tracking categories. Non-interactive includes the Support database [18], consisting of articles written by users. Interactive lists forums, mailing Lists, and IRC channels, all of which have medium to high traffic. Bug tracking is where you'll find the project's Bugzilla. Although mostly self-contained, openSUSE's support and documentation generally maintains high standards.

Ubuntu

Ubuntu is the source of many small distributions. Perhaps for that reason, it maintains basic to intermediate documentation at a separate URL from the distribution [19]. This documentation may also come in handy for those who do not care to sort through all the links on the Debian site. Only the Long Term Support (LTS)

releases are consistently documented. Server and developer information are on a separate site [20] (Figure 7).

Forums for Ubuntu are tightly organized, with separate ones for different flavors, such as Kubuntu and Lubuntu; others for specific tasks, among them security, Wine, and System76 support (for a company that offers Ubuntu as a choice on its computers); and ones for specific interests within the Ubuntu project [21]. Sadly, the independent *Full Circle Magazine* is no longer published, although back issues continue to be available. ∎∎∎
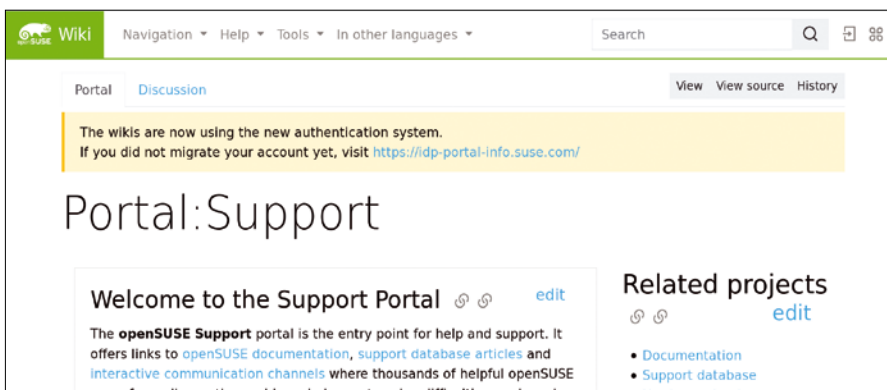
Figure 6: Like many distros, openSUSE emphasizes forums and IRC channels for support.
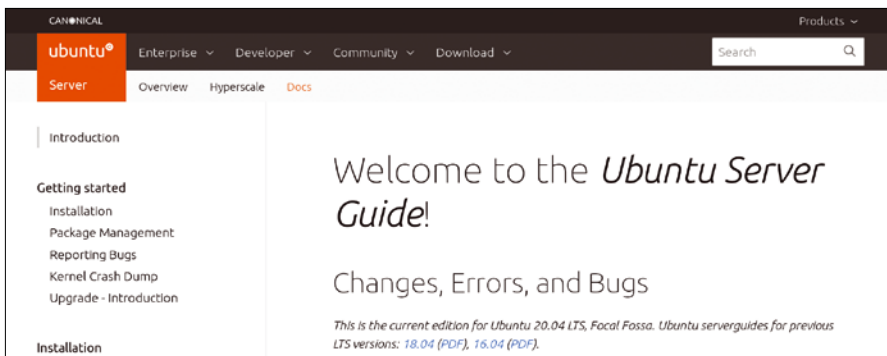
Figure 7: Ubuntu has moved its server documentation to a separate site.

Info

[1] Arch Linux documentation: *https://wiki.archlinux.org/*

[2] Arch Linux forums: *https://bbs.archlinux.org/*

[3] Bodhi wiki: *https://www.bodhilinux.com/w/wiki/*

[4] Bodhi forums: *https://www.linuxquestions.org/questions/bodhi-92/*

[5] Debian developers: *https://www.debian.org/devel/*

[6] Debian packages: *https://www.debian.org/distrib/packages*

[7] Debian documentation: *https://www.debian.org/doc/*

[8] Debian forums: *http://forums.debian.net/*

[9] Fedora documentation: *https://docs.fedoraproject.org/en-US/docs/*

[10] Ask Fedora: *https://ask.fedoraproject.org/*

[11] *Fedora Magazine*: *https://fedoramagazine.org/*

[12] Linux Mint guides: *https://linuxmint.com/documentation.php*

[13] Linux Mint Cinnamon development guide: *https://linuxmint-developer-guide.readthedocs.io/en/latest/cinnamon.html*

[14] Linux Mint forums and IRC: *https://forums.linuxmint.com/*

[15] openSUSE documentation priority: *https://en.opensuse.org/Portal:Documentation*

[16] openSUSE how-tos: *https://en.opensuse.org/Category:SDB:HOWTOs*

[17] openSUSE Support portal: *https://en.opensuse.org/Portal:Support*

[18] openSUSE Support database: *https://en.opensuse.org/Portal:Support_database*

[19] Ubuntu documentation: *https://help.ubuntu.com/*

[20] Ubuntu server documentation: *https://ubuntu.com/server/docs*

[21] Ubuntu forums: *https://ubuntuforums.org/*

Advanced command-line navigation

Beyond cd

For decades, cd has been a Linux workhorse, but modern drives sometimes call for more efficient navigation tools. Read on for alternatives for advanced command-line navigation. *By Bruce Byfield*

Change directory (`cd`) is one of the first commands that Linux users discover at the command line. It is so basic that it has only a few options. An example of the Unix philosophy that a command should do one thing and do it well, `cd` has survived for decades with no change of which I am aware. Still, it was designed for a simpler time, when storage was scarce. On modern drives, it can involve a lot of typing of paths, especially if your directory structure is more than one level deep.

So what are the alternatives? The answers are surprisingly numerous, ranging from those built into the shell, to more recent options that may require extra setup.

Built-In Alternatives

The shell has several ways to make navigation more efficient:

- `cd`'s own shortcuts: Typing `cd ..` moves you to the current directory's parent directory. A tilde (`~`) moves

Author

Bruce Byfield is a computer journalist and a freelance writer and editor specializing in free and open source software. In addition to his writing projects, he also teaches live and e-learning courses. In his spare time, Bruce writes about Northwest coast art (*http://brucebyfield.wordpress. com*). He is also co-founder of Prentice Pieces, a blog about writing and fantasy at *https://prenticepieces.com/.*

you to your home directory and a hyphen (`-`) to the previous directory (Figure 1).

- The print working directory (`pwd`) command can be used to change elements of the directory name using the structure `cd ${PWD/OLD/NEW}`. This structure is mostly useful for changing between directories in the same level of the directory hierarchy. It can also be effective in scripting (Figure 2).

- The Bash history can be navigated with arrow keys or by specifying the number of an item in the history with `!NUMBER`. If you have recently switched to a directory, this may be an alternative.

- Pressing the Tab key produces possible completions. Eventually as you type, fewer completions are left until only one remains, and it is automatically entered.

- Symbolic links can have a shorter name than their original reference. Creating a series of symbolic links as children of your home directory creates a do-it-yourself navigation structure with minimal typing.

While these built-in alternatives improve on `cd`'s efficiency, the alternatives listed below have more functionality and are often more efficient. For some, these alternatives' main shortcoming is that they require some configuration to be useful.

autojump

`autojump` [1] records the directories you switch to using `cd` and creates a database of shortcuts. It requires a somewhat complicated setup. In Fedora and related distributions, you will need to enable the Extra Packages for Enterprise Linux (EPEL) repository that includes `autojump` [2]. Most distributions enable

```
bb@nanday:~/Downloads/Sorcery-and-Steam$ cd ..
bb@nanday:~/Downloads$ cd ~
bb@nanday:~$ cd -
/home/bb/Downloads
bb@nanday:~/Downloads$ ▮
```

Figure 1: The `cd` command contains several shortcuts.

```
bb@nanday:~/fonts$ cd ${PWD/fonts/music}
bb@nanday:~/music$ ▮
```

Figure 2: `pwd` can be a quick way to jump to another directory.

Lead Image © artqu, 123rf.com

```
bb@nanday:~/music$ j --stat
14.1:    /home/bb/Downloads
24.5:    /home/bb/autojump
24.5:    /home/bb/projects
46.5:    /home/bb/music
75.0:    /home/bb/fonts
-------------------------------------------
184:     total weight
5:       number of entries
46.48:   current directory weight

data:    /home/bb/.local/share/autojump/autojump.txt
```

Figure 3: `autojump` **maintains a database of directories, assigning each a priority according to how many times you change into it.**

`autojump` upon installation, but before using Debian and its derivatives you must run the following command:

```
source /usr/share/autojump/autojump.sh ⤴
  on startup
```

To permanently enable `autojump`, you can add the same command to `~/.bashrc` so that `autojump` is available whenever you open a Bash shell.

Beware: If you are using the Zsh shell in any distribution, you will need to find the Zsh version of `autojump`. Without this addition, `autojump` will not work in Debian. You also have the option of installing from source by cloning from a user account with the command

```
git clone git://github.com/joelthelion/⤴
  autojump.git
```

and then changing the permissions on the file `install.py` and running it as root.

Once `autojump` is set up, it gradually adds directories to its database (`~/.local/share/autojump/`) as you navigate the command line. Alternatively, you can add directories manually with the option `--add DIRECTORY` (`-a`). Either way, when a directory is added to `autojump`, you can switch to it using either the command `autojump` or simply `j`. Since you only have to type the directory name and not the complete path, you can save keystrokes, especially if you have a deep directory structure. For instance, instead of typing

```
cd /home/bb/work/journalism/2020/August
```

I could type simply

```
j August
```

Should you choose, you can also open a directory in the default file manager of your desktop environment using the command `jo DIRECTORY`. If a directory is not a top-level one, then the command is `jc DIRECTORY`. Depending on the speed of your machine, the desktop file manager may take a few seconds to open. You can also use `--complete` to enable tab completion in your current session.

You can view the database with the option `--stat` (Figure 3). You will notice that directories are weighted according to how many times you switch to them. You can use the options `--increase WEIGHT DIRECTORY` (`-i`) and `--decrease WEIGHT DIRECTORY` (`-d`) to help eliminate any confusion between similarly named directories and `--purge` to remove paths that have been removed. Looking at the `autojump_errors` file can help you detect duplicate path names.

`autojump` is unable to work with files that start with a hyphen. Otherwise, after it is set up, `autojump` is a powerful navigation tool.

cdargs

`cdargs` [3] is similar to `autojump`, except that its documentation refers to directory bookmarks. Similarly to `autojump`, `cdargs` installs by evoking one of the scripts provided in its package, either in the current session with the command

```
source /usr/share/doc/cdargs/examples/⤴
  cdargs-bash.sh
```

or permanently by adding the same command to your `bash.rc` file.

You can build a navigation system by changing to a directory and running the command `ca`, which will create a bookmark with the same name as the directory, or `ca BOOKMARK`, which can provide a shorter name. Alternatively, you can run `cv` and use `a` to add the current directory by its full name, `d` to delete the highlighted directory, and `e` to edit the list in your default text editor. You can close any list window with `q`.

To use `cdargs` to navigate, use the command `cdb BOOKMARK`. If you do not specify a directory, `cdb` opens a list of directories from which to choose. You can also enter `cdb Lc` to select by highlighting an entry on a list of directories and pressing the Enter key (Figure 4).

mc

Midnight Commander (`mc`) [4] is a clone of Norton Commander, a DOS file manager that was popular in the early 1990s before the rise of Windows 3.1. Like Norton Commander, `mc` provides a simple graphical interface that is navigated entirely by keys. The main difference is that `mc` has been kept mostly up to date and includes such features as support

```
   [.        ]  /home/bb/projects
0  [DWL      ]  /home/bb/projects/DWL
1  [Freda-Di]  /home/bb/projects/Freda-Diesing-Book
2  [IR-card-]  /home/bb/projects/IR-card-game
3  [creative]  /home/bb/projects/creative
4  [ebooks  ]  /home/bb/projects/ebooks
5  [editing-]  /home/bb/projects/editing-services
6  [music   ]  /home/bb/projects/music
7  [prentice]  /home/bb/projects/prentice-pieces
8  [ruth-poe]  /home/bb/projects/ruth-poetry
9  [shadow-o]  /home/bb/projects/shadow-of-the-lion
   [test    ]  /home/bb/projects/test

B: /home/bb/projects
```

Figure 4: **You can choose** `cdarg`**'s bookmarks from a list.**

Figure 5: `mc` **provides a file manager for the command line.**

Figure 6: `tree` **gives you another option for displaying directories.**

for multiple encoding and the ability to access remote files via SSH. However, the occasional anachronism lingers, such as an `undelete` command, which supports only ext2 filesystems, and support for FTP.

In many ways, `mc` is similar to many desktop file managers. It features a two-pane interface and a sub-shell for executing commands. The two panes and the menus can be navigated either with keys or the mouse, but the directory tree and dialogs work only with the mouse. Common commands are accessed from the function keys listed at the bottom of the interface, while other commands, including ones to find and compare files specifically within `mc`, are available from the Command menu. In addition, custom commands can be

added to `mc` as needed. `mc` itself can be extensively customized from the Options menu, with support for different skins, as well as the notification types displayed, panel layouts, and keyboard shortcuts (Figure 5).

`mc` is best suited to routine commands, as well as new or occasional users of the command line for whom a graphical interface is familiar and reassuring. Users more familiar with the command line may find `mc` restrictive and chafe at the replacement of `ls` with scrolling. For those familiar with the Vim text editor, an alternative might be Vifm [5].

Other Tools

In addition to these tools, readers might also be interested in commands

like `readline`, which is designed for navigating a single line at the prompt, and the various `goto` commands in scripting languages.

In particular, any of the navigation tools might be paired with `tree` [6], a replacement for `ls` that displays directories and files in the traditional tree structure and provides a visual display that the command line generally lacks (Figure 6). `tree` includes numerous options, such as display only directories or full files. However, with 3TB drives common today, `tree` should generally be piped through the `less` command to reduce scrolling.

`cd` itself has survived for several decades and shows no immediate signs of becoming obsolete. However, in some circumstances, it is no longer as efficient as it once was. When you run up against `cd`'s limitations, it is reassuring to know that alternatives exist. ∎∎∎

Info

[1] autojump:
    https://github.com/wting/autojump

[2] EPEL Repository: *https://www.tecmint.com/install-epel-repository-on-centos/*

[3] cdargs:
    https://directory.fsf.org/wiki/Cdargs

[4] mc: *https://midnight-commander.org/*

[5] Vifm: *https://vifm.info/*

[6] tree: *https://linux.die.net/man/1/tree*

The sys admin's daily grind: Formatting JSON for readability

Gimme Output

Armed with just json.tool and jq, Charly preps the JSON data delivered by his Philips Hue bridge so that even humans can read it – an essential step towards improving the usability of his home automation system. *By Charly Kühnast*

Some time ago [1], I briefly talked about how I use the measured values from a lux sensor to control a Hue lighting system. In redecorating my hallway, I added some new lights and removed others. I couldn't remember the light sources' IDs, so I asked the Hue bridge to dump the configuration. What I got was hardcore JSON, but unfortunately not in a human-readable format (Figure 1).

There are plenty of tools to make JSON readable. I started with what I already had in place, json.tool, which the Python installation had dumped on my hard disk. I wrote its neatly formatted output to a file in /tmp for further processing (Listing 1).

However, I didn't need the data in Python; I wanted to use the output in a small Bash script. This prompted me to continue processing with jq [2]. The tool claims to be a kind of sed, awk, and grep for JSON. I wanted to find out the current IDs for my light sources, as well as discover their plain text names. Thanks to jq, a tiny script (Listing 2) is all

it takes to do this. When I ran it, it gave me the output shown in Listing 3.

The reason why IDs 1 to 3 do not exist is because I dismantled the lights and removed them from the Hue app. The Hue bridge has no reason to re-sort the remaining IDs – which is fortunate, because that would cause total chaos every time I changed a lamp. As jq once again underlines: The shorter a tool's name is; the more important and powerful it is. Lights on! ∎∎∎

Listing 2: Light Source Script

```
01 #! /bin/bash
02 WDIR=/usr/local/shellscripts/lux
03 TMPDIR=/tmp
04 HUEBRIDGE=10.0.0.10
05 USER=w25-4kqL7d
06
07 for i in $(seq 1 20); do
08   echo "trying light $i";
09   LAMPTEST=$(jq -r -M ".lights.\"$i\".name" /$TMPDIR/hue.all);
10   if [ "$LAMPTEST" != "null" ]; then
11     echo "Light $i exists: $LAMPTEST"
12   fi
13 done
```

Listing 1: JSON Query

```
$ curl --request GET 10.0.0.10/api/w25-4kqL7d|python -m json.tool > /tmp/hue.all
```

{"lights":{"4":{"state":{"on":false,"bri":2,"alert":"select","mode":"homeautomation",
"reachable":false},"swupdate":{"state":"noupdates","lastinstall":"2020-04-17T13:17:40
"},"type":"Dimmable light","name":"WZ_Decke_Schreibtisch","modelid":"LWB010","manufac
turername":"Signify Netherlands B.V.","productname":"Hue white lamp","capabilities":{
"certified":true,"control":{"mindimlevel":5000,"maxlumen":806},"streaming":{"renderer
":false,"proxy":false}},"config":{"archetype":"classicbulb","function":"functional","
direction":"omnidirectional","startup":{"mode":"custom","configured":true,"customsett
ings":{"bri":1}}},"uniqueid":"00:17:88:01:02:b2:78:3c-0b","swversion":"1.50.2_r30933"
,"swconfigid":"1D8EE00F","productid":"Philips-LWB010-1-A19DLv3"},"5":{"state":{"on":f
alse,"bri":2,"alert":"select","mode":"homeautomation","reachable":false},"swupdate":{
"state":"noupdates","lastinstall":"2020-04-17T13:17:42"},"type":"Dimmable light","nam
e":"WZ_Decke_Sofa","modelid":"LWB010","manufacturername":"Signify Netherlands B.V.","
productname":"Hue white lamp","capabilities":{"certified":true,"control":{"mindimleve
l":5000,"maxlumen":806},"streaming":{"renderer":false,"proxy":false}},"config":{"arch
etype":"classicbulb","function":"functional","direction":"omnidirectional","startup":
{"mode":"custom","configured":true,"customsettings":{"bri":1}}},"uniqueid":"00:17:88:
01:02:b1:65:c7-0b","swversion":"1.50.2_r30933","swconfigid":"1D8EE00F","productid":"P
hilips-LWB010-1-A19DLv3"},"6":{"state":{"on":false,"bri":1,"hue":7676,"sat":199,"effe
ct":"none","xy":[0.5016,0.4151],"ct":443,"alert":"select","colormode":"xy","mode":"ho
meautomation","reachable":true},"swupdate":{"state":"noupdates","lastinstall":"2020-0
3-27T13:15:35"},"type":"Extended color light","name":"ColorWall1","modelid":"4090231P
9","manufacturername":"Signify Netherlands B.V.","productname":"Hue color wall","capa
bilities":{"certified":true,"control":{"mindimlevel":100,"maxlumen":900,"colorgamutty
pe":"C","colorgamut":[[0.6915,0.3083],[0.1700,0.7000],[0.1532,0.0475]],"ct":{"min":15
3,"max":500}},"streaming":{"renderer":true,"proxy":true}},"config":{"archetype":"wall
shade","function":"functional","direction":"vertical","startup":{"mode":"custom","con
figured":true,"customsettings":{"bri":128,"xy":[0.5018,0.4156]}}},"uniqueid":"00:17:8
8:01:06:f8:88:f7-0b","swversion":"1.65.9_hB3217DF","swconfigid":"FA142228","productid

Figure 1: Unreadable: Pure JSON.

Listing 3: Script Output

```
Light 4 exists: LR_ceiling_desk
Light 5 exists: LR_ceiling_sofa
Light 6 exists: ColorWall1
Light 7 exists: Hall_1F
Light 8 exists: Hall_2F
Light 9 exists: Hall_3F
Light 10 exists: LRTinkeringCorner
```

Author

Charly Kühnast manages Unix systems in a data center in the Lower Rhine region of Germany. His responsibilities include ensuring the security and availability of firewalls and the DMZ.

Info

[1] "Charly's Column – Hue and Rasp Pi" by Charly Kühnast, *Linux Magazine*, issue 218, January 2019, p.39, *https://www.linux-magazine.com/Issues/2019/218/Hue-and-Rasp-Pi/(language)/eng-US*
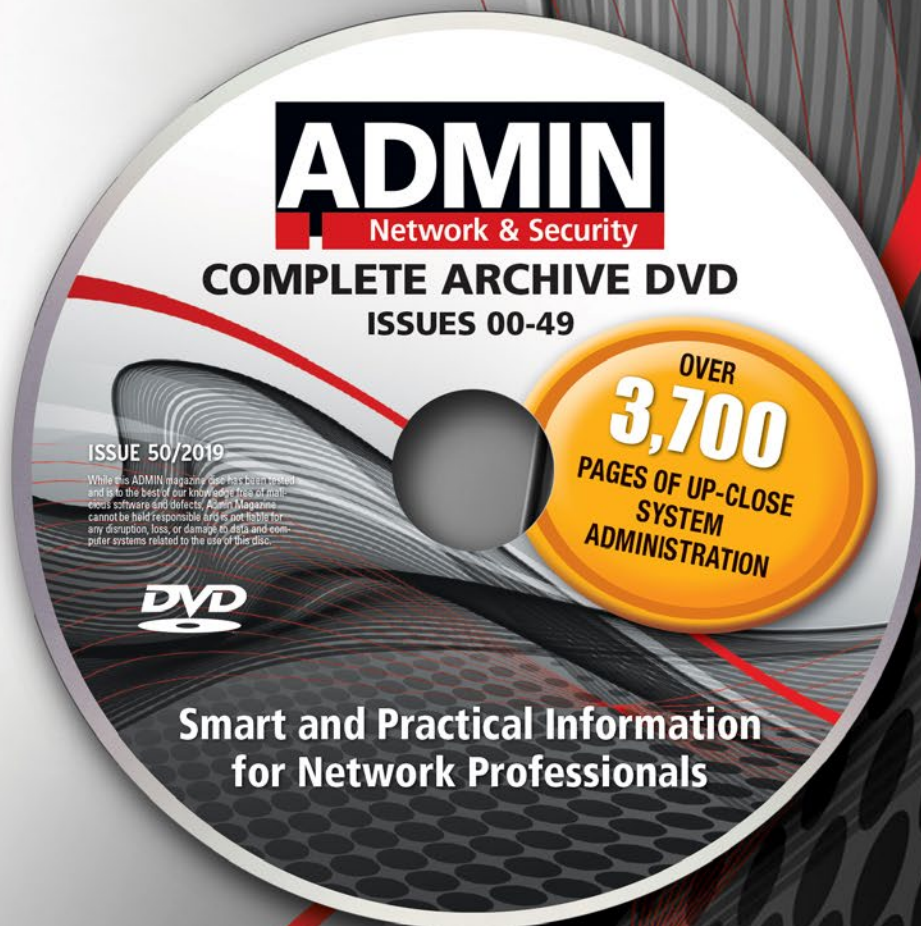
[2] jq: *https://stedolan.github.io/jq/*

Reliable videoconferencing

Conference Call

If you are looking for an alternative to commercial videoconferencing platforms, Jitsi offers an open source solution that lets you build and deploy online videoconferences. *By Chris Binnie*

Halfway through a phone conversation amid the doom and gloom of the global pandemic, someone complained about commercial online videoconferencing services creaking at the seams due to the massive increase in demand.

This led me to think about open source video conferencing options. After a little bit of reading, I discovered Jitsi [1], a truly impressive open source project for building and deploying secure videoconference services. Jitsi, courtesy of 8x8 Inc. [2], has an Apache2 license and includes Jitsi Meet [3], a free video conferencing platform, and Jitsi VideoBridge, a Selective Forwarding Unit (SFU) that powers multiperson videoconferencing. For an overview of the project, visit their website [4] or check out their GitHub page [5].

Jitsi offers secure and scalable videoconferences, either as a standalone app or embedded into your own web application. Jitsi has been around since 2003 in one incarnation or another, with continual fine tuning over recent years.

You do not need an account to start a video call with Jitsi, and it is very browser-friendly and mobile-compatible. Jitsi also allows screen sharing,

voting with your hand, and online chat. In this article, I'll show you how to use Jitsi to create a fully encrypted, open source, free-to-use, videoconferencing service.

Getting Started

I'll walk you through what you need to get Jitsi up and running. Also, I'll assist in switching off as many network ports as possible so that you're only exposing the HTTPS port to the public, and I'll provide some fine-tuned firewall rules. To get an insight into all the features Jitsi offers, I will install the

full suite of Jitsi software. You will be pleasantly surprised at how little the disk footprint is.

For my videoconferencing server, I'll use DigitalOcean [6], which refers to servers as "droplets." When creating the droplet, I'll ask the cloud platform to install Ubuntu 18.04 (Bionic Beaver) as it has a shelf life until April 2028. I'll assume that you can create a DigitalOcean account on your own. Setting up a droplet manually is easy – if you are new to DigitalOcean, visit their website for more information on setting up a droplet [7].

Listing 1: Confirming Operating System

```
NAME="Ubuntu"

VERSION="18.04.4 LTS (Bionic Beaver)"

ID=ubuntu

ID_LIKE=debian

PRETTY_NAME="Ubuntu 18.04.4 LTS"

VERSION_ID="18.04"

HOME_URL="https://www.ubuntu.com/"

SUPPORT_URL="https://help.ubuntu.com/"

BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"

PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"

VERSION_CODENAME=bionic

UBUNTU_CODENAME=bionic
```

Lead Image © Dan Barbalata, 123RF.com

After logging in over SSH, run

```
apt update; apt upgrade -y
```

followed by a reboot to update the operating system software. After running the following command:

```
$ cat /etc/os-release
```

you will get the output shown in Listing 1, confirming that the server is running the correct Ubuntu version.

The next trick is to install Apache2 (or NGINX if you prefer). During installation, Jitsu will figure out which web server you are using and then work via that server. Install the Apache2 web server with:

```
$ apt install apache2
```

After entering the droplet's IP address into a browser, you will see that Apache2 is running as expected (Figure 1).

Next, I'll follow the installation instructions from Jitsi's website [8].

To ensure that I'm connecting to the right repository, as the root user, I'll set up the key for the Jitsi repository with the following:

```
$ wget -qO ⤵
  - https://download.jitsi.org/⤵
  jitsi-key.gpg.key | apt-key add -
OK
```

Then, I'll open up a file called `/etc/apt/sources.list.d/jitsi-stable.list` and save the following line to it:

```
deb https://download.jitsi.org stable/
```

After that, I'll do a quick update as shown in Listing 2.

As you can see from the output in Listing 2, the Jitsi repo has been connected to as hoped and accessed without complaint following the key that I installed.

Listing 2: Connection to Jitsi Repository

```
$ apt update

Hit:1 http://security.ubuntu.com/ubuntu bionic-security InRelease

Get:2 http://mirrors.digitalocean.com/ubuntu bionic InRelease [242 kB]

Get:3 https://download.jitsi.org stable/ InRelease [2415 B]

Hit:4 http://mirrors.digitalocean.com/ubuntu bionic-updates InRelease

Hit:5 http://mirrors.digitalocean.com/ubuntu bionic-backports InRelease

Get:6 https://download.jitsi.org stable/ Packages [36.4 kB]

Fetched 281 kB in 1s (363 kB/s)

Reading package lists... Done

Building dependency tree

Reading state information... Done

All packages are up to date.
```

Installing Jitsi

I'm now ready to install the videoconferencing suite, barring some firewall tweaking. The commands in Listing 3 install the full package.

As expected, the installation involves a reasonably large number of files, but it only takes about 215MB of extra space on my server. You might notice that the *x11-common* package will be installed; usually this package wouldn't be used on a headless server driven by the command line. Consequently, you might want to have this server dedicated to Jitsi.

The *jitsi-meet* package installs the whole suite, which I will do to show all of Jitsi's features. You also can just install the individual components you need as follows:

```
$ apt install jitsi-videobridge
$ apt install jicofo
$ apt install jigasi
```


Figure 1: Happiness is Apache2 serving HTML as expected.

Listing 3: Installing Jitsi

```
$ apt install jitsi-meet

ca-certificates-java fontconfig-config fonts-dejavu-core java-common jicofo jitsi-meet jitsi-meet-prosody jitsi-meet-web
  jitsi-meet-web-config jitsi-videobridge libavahi-client3 libavahi-common-data libavahi-common3 libcups2 libfontconfig1
  libjpeg-turbo8 libjpeg8 liblcms2-2 libnspr4 libnss3 libpcsclite1 libxi6 libxrender1 libxtst6 lua-bitop lua-event lua-expat
  lua-filesystem lua-sec lua-socket lua5.1 openjdk-8-jre-headless prosody x11-common
```

Listing 4: Generating a Self-Signed Certificate

```
Running hooks in /etc/ca-certificates/update.d...
done.
done.
Processing triggers for ureadahead (0.100.0-21) ...
Processing triggers for libc-bin (2.27-3ubuntu1) ...
```

Listing 5: Let's Encrypt Installation Successful

```
 - Congratulations! Your certificate and chain have been saved at:
   /etc/letsencrypt/live/call.chrisbinnie.tld/fullchain.pem
   Your key file has been saved at:
   /etc/letsencrypt/live/call.chrisbinnie.tld/privkey.pem
   Your cert will expire on 2020-06-22. To obtain a new or tweaked
   version of this certificate in the future, simply run certbot-auto
   again. To non-interactively renew *all* of your certificates, run
   "certbot-auto renew"
 - If you like Certbot, please consider supporting our work by:

   Donating to ISRG / Let's Encrypt:   https://letsencrypt.org/donate
   Donating to EFF:                     https://eff.org/donate-le

Configuring apache2
```

Listing 6: iptables.sh

```
01 # Clear old config first
02 iptables -P INPUT ACCEPT
03 iptables -P FORWARD ACCEPT
04 iptables -P OUTPUT ACCEPT
05 iptables -t nat -F
06 iptables -t mangle -F
07 iptables -F
08 iptables -X
09
10 # Drop inbound and forwarding but allow outbound
11 iptables -P INPUT DROP
12 iptables -P FORWARD DROP
13 iptables -P OUTPUT ACCEPT
14
15 # Allow localhost in/out
16 iptables -A INPUT -i lo -j ACCEPT
17 iptables -A OUTPUT -o lo -j ACCEPT
18
19 # Allow established traffic to flow
20 iptables -A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
21
22 # Apparently Jitsi can use these ephemeral ports so allow them inbound
23 iptables -A INPUT -p udp --dport 10000:20000 -j ACCEPT
24
25 # Enable HTTPS access
26 iptables -A INPUT -p tcp --dport 443 -j ACCEPT
27
28 # Allow HTTP for Let's Encrypt temporarily
29 iptables -A INPUT -p tcp --dport 80 -j ACCEPT
30
31 # Change the default port in /etc/ssh/sshd_config from 22 after testing
32 iptables -A INPUT -p tcp --dport 2002 -j ACCEPT
```

Upon launching the *jitsi-meet* package, you will be asked for a hostname. Choose a sensible one. Assuming you own a domain name, create a DNS name (an A record) for your server's IP address to save time later. I've chosen this hostname:

```
call.chrisbinnie.tld   IN   A  ⤵
  123.123.123.123
```

Enter your hostname in the pop-up box and then choose *Generate a new self-signed certificate* on the next screen. (You will get a chance later to obtain a Let's Encrypt certificate.)

At this stage, screeds of data are flying up the terminal as the Jitsi software runs through its installation. Listing 4 shows the tail end of the output without errors before returning you to a prompt.

Next Jitsi will auto-configure a Let's Encrypt certificate for free and associate it with your hostname. You need to wait for your DNS to become available (if your DNS isn't ready, then just run it again when the DNS is available) to run a sophisticated Let's Encrypt script. To confirm that your DNS is live, run this redacted command:

```
$ host -v -t a call.chrisbinnie.tld
  ns1.nameserver.tld
```

and then run the Let's Encrypt script:

```
$ /usr/share/jitsi-meet/scripts/⤵
  install-letsencrypt-cert.sh
```

In this brief process, you're prompted for an email address. Then, as if by magic, certificates are mysteriously installed, taking all the configuration pain away (Listing 5).

You can renew the certificate easily in the future by running the command again and look for the output as shown in Listing 5.

Firewall Rules

Your next step in to set up your firewall rules.

I won't lock down your SSH port to a specific IP address in this example, because I don't want you locked out mid-configuration. You can use iptables or even TCP Wrappers to lock down your SSH yourself. Once testing is finished, you also need to change the SSH port and harden your SSH daemon (if you wish).

In order to configure iptables from Netfilter, I'll first create a simple script, /root/iptables.sh (Listing 6), and then make it executable with the command

Listing 7: Firewall Rules Persist

```
Chain INPUT (policy DROP 23 packets, 1768 bytes)

 pkts bytes target  prot opt in  out  source      destination

  174 24898 ACCEPT  all  -- lo  *    0.0.0.0/0 0.0.0.0/0

   65  7817 ACCEPT  all  -- *   *    0.0.0.0/0 0.0.0.0/0  ctstate
RELATED,ESTABLISHED

    0     0 ACCEPT  udp  -- *   *    0.0.0.0/0 0.0.0.0/0  udp dpts:10000:20000

    0     0 ACCEPT  tcp  -- *   *    0.0.0.0/0 0.0.0.0/0  tcp dpt:443

    0     0 ACCEPT  tcp  -- *   *    0.0.0.0/0 0.0.0.0/0  tcp dpt:80

    4  1095 ACCEPT  tcp  -- *   *    0.0.0.0/0 0.0.0.0/0  tcp dpt:22


Chain FORWARD (policy DROP 0 packets, 0 bytes)

 pkts bytes target  prot opt in  out  source      destination


Chain OUTPUT (policy ACCEPT 62 packets, 9148 bytes)

 pkts bytes target  prot opt in  out  source      destination

  174 24898 ACCEPT  all  -- *   lo   0.0.0.0/0 0.0.0.0/0
```

```
chmod +x /root/iptables.sh
```

so that I have a backup of my firewall rules.

These firewall rules are just an example, so you might need to adjust them according to your needs. You can always log in via the DigitalOcean console (in your browser) if you get stuck here and lose SSH access.

To run the script in Listing 6 enter:

```
$ /root/iptables.sh
```

To verify that the firewall rules are in place, use:

```
$ iptables -nvL
```

To make sure your custom iptables firewall rules persist after a reboot, install the very handy *iptables-persistent* package with:

```
$ apt install iptables-persistent
```

Then, choose *yes* to save IPv4 rules and select *no* for IPv6 rules as there aren't any. A quick reboot is needed to check that you can still get back into the server by SSH.

If I run the `iptables -nvL` command again to verify that the firewall rules have persisted, I will see the output displayed in Listing 7.

In short, the rules in Listing 7 mean that outbound traffic is allowed. How-ever, inbound traffic is only permitted if it's part of a conversation already taking place and HTTP is open (close this soon after you finish testing since the script leaves it open, but note that your LetsEncrypt certificate refreshes need this port open) as well as HTTPS and SSH ports (which you should lock down yourself and change the port number). UDP ports are also enabled for ephemeral connections in the 10000-20000 range (this may not be required).

Start a Meeting

Now, for the moment of truth. To see if I can access my videoconferencing suite, I enter the following URL in my browser: *https://call.chrisbinnie.tld*. Success: Jitsi Meet's welcome screen appears with a clean, easy on the eyes design (Figure 2).

In the scrolling text under the *Start a new meeting* box, random words appear to prevent someone from immediately guessing your meeting's name and joining without an invite. Remember this software is so accessible that you don't even need to sign up for an account. Note, however, that it's easy to add a password to the meeting to prevent unwelcome visitors.

Once you start a meeting, you can see Jitsi flex its muscles.

If you start Jitsi with your camera switched off, you will see the screen shown in Figure 3. On the bottom left, you'll find a screen sharing icon, a raise your hand to vote icon (or pay attention to me icon), and an icon to start text-based chats. In the middle at the bottom of the screen, there's the familiar mute/unmute, hang up, and start/stop video buttons. To the bottom right, a settings menu lets you do things like blurring the background so that your camera focuses on you for a bit more privacy (currently in beta, so use with caution – my CPU didn't like this feature much).

If you enter your name and an email address associated with Gravatar [9], then you will be associated with a graphical avatar.
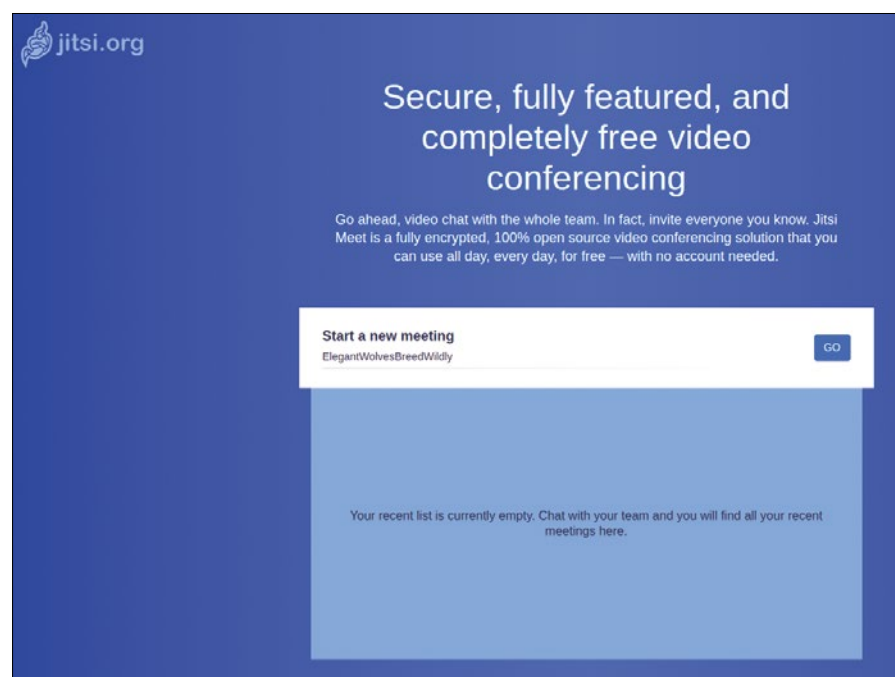
Figure 2: A warm welcome to Jitsi's videoconferencing suite.
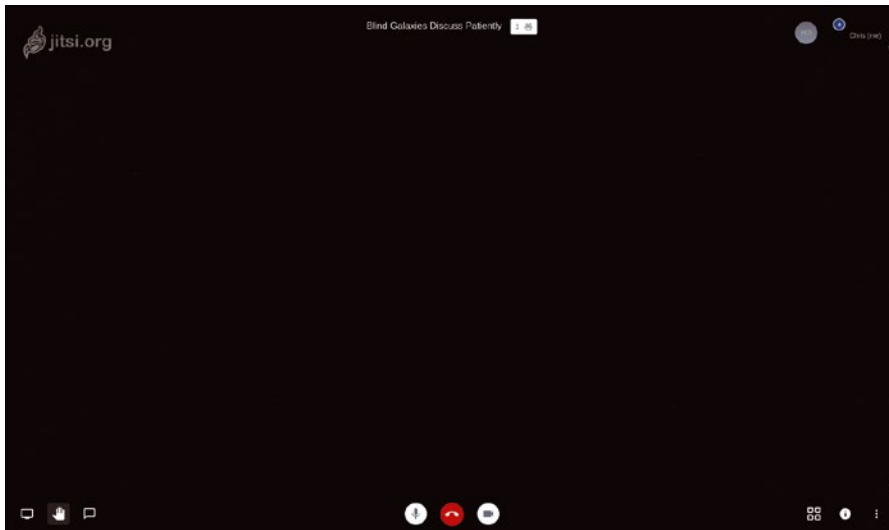
Figure 3: Starting a meeting in Jitsi with the camera disabled.

Another setting lets you choose different microphones and webcams on your system. I have two of each; using WebRTC [10], both are available in Google Chrome without issue. Although Mozilla Firefox should work in the same way, there've been a few reports of issues, but I've had no issues with Chrome and Firefox. With Firefox I had the most success with screen sharing. If in doubt, clear your browser cache and then restart the browser; that seems to free up WebRTC resources again.

Join In

It's also remarkably easy to join video calls with standard telephony. According to the documentation, it's quite possible to connect to your server via multiple international telephone numbers if you set up Jigasi [11] (the Jitsi gateway to SIP) to use a SIP provider. (See [12] for an explanation of the difference between VoIP and SIP.)

On Jitsi Meet, a Chrome pop-up will encourage you to install a Chrome Extension to integrate Google Calendar and Office 365 (Figure 4).

For iOS users, navigate to Jitsi Meet on IOS [13], where you can install a client on Apple devices. According to the website, it's available for iPhone, iPad,

and Apple Watch, but sadly I can't test on these devices.

I can however confirm that the Android app [14] works. You don't need to install a server of your own either; it just uses Jitsi Meet automatically. The app is slick, simple, and fast and connects to my server (*https://call.chrisbinnie.tld*) instantly if I alter the settings under *Server URL*.

Jitsi also offers packages for F-Droid [15], React Native, and Electron.

The End Is Nigh

With Jitsi, there are no artificial user limits. Instead, bandwidth and server resources ultimately determine the number of users.

On DigitalOcean, I purposely chose a $5/month droplet for testing. I have had a few streams connected to it at once with no noticeable additional load. I've been monitoring the server as I've used it via SSH. If you notice drop-outs or other issues, then try getting your fellow participants to drop down from full HD streaming. Of course, you will need significant bandwidth and more servers for multiple chats in an organization, but the cloud makes light work of the required flexibility.

If you start creaking at the seams with just one server instance, you can scale a

Jitsi installation as described in an article by Jan Doberstein [16]. The scaling seems fairly intuitive, once you know how. Doberstein says that the real resource hog is *jitsi-videobridge*, which you should scale first. Since this issue seems likely to arise at some point, I encourage you to read through Doberstein's article to find the fix and learn more about the software's innards. The article is current and any references to the latest release should apply.

Since creating my Jitsi server, I have made sure that I closed down the HTTP port and added more security to the default SSH installation, changing the port by locking it to my IP addresses. As an extra assurance, the iptables' default deny policy keeps out everything but SSH and HTTPS except for traffic that originates internally and wants to go outbound. I've also added simple scripts to alert me when updates are available so I don't miss them if I don't log in for a few days.

If you are looking for an alternative to overloaded commercial videoconferencing services, give Jitsi a try. I hope you will enjoy Jitsi as much as I have. ∎∎∎

Info

[1] Jitsi: *https://jitsi.org*

[2] 8x8 Inc.: *https://www.8x8.com*

[3] Jitsi Meet: *https://meet.jit.si*

[4] What Is Jitsi?: *https://jitsi.org/what-is-jitsi*

[5] Jitsi GitHub page: *https://github.com/jitsi/jitsi-meet*

[6] DigitalOcean: *https://www.digitalocean.com*

[7] Setting up a droplet: *https://www.digitalocean.com/docs/droplets/how-to/create*

[8] Installing Jitsi: *https://jitsi.org/downloads/ubuntu-debian-installations-instructions*

[9] Gravatar: *https://en.gravatar.com*

[10] WebRTC: *https://webrtc.org*

[11] Jigasi: *https://github.com/jitsi/jigasi*

[12] SIP vs. VOIP: *https://www.sip.us/blog/latest-news/sip-vs-voip-whats-difference*

[13] Jitsi on iOS: *https://apps.apple.com/us/app/jitsi-meet/id1165103905*

[14] Jitsi on Android: *https://play.google.com/store/apps/details?id=org.jitsi.meet&hl=en*

[15] Jitsi on F-Droid: *https://f-droid.org/en/packages/org.jitsi.meet/*

[16] Scaling Jitsi: *https://jalogisch.de/2020/scale-jitis-make-videochat-more-efficient*
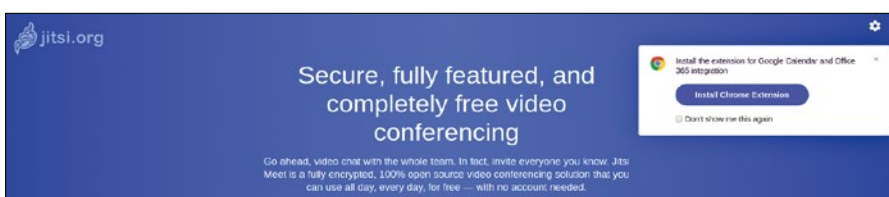
Figure 4: You can join a Jitsi Meet call without installing anything. However, you're encouraged to integrate with Google Calendar and Office 365.

A package search engine for Linux repositories

Package Monitor

The Repology web app provides up-to-date information on software archives, versions, package maintainers, and more. And it goes beyond the boundaries of Linux. *By Ferdinand Thommes*

Most of us spend a significant amount of time in front of our computers searching for information – whether in the browser via a search engine, in the filesystem with commands like `find` and `locate`, or with graphic tools. One of the commands most frequently used with us, `whohas`, determines which distributions have which version of a specific package in their archives. Using this command can be frustrating: `whohas` does not support all distributions, and not all distributions' repositories can be reached at the time of a query.

In a search for alternatives, I came across a genuine treasure chest, Repology [1], a web app that lists the software packages of many distributions. It retrieves this data hourly directly from the distributions' repositories. Repology links together a versatile set of information about the individual packages. At the time of writing, Repology covered more than 120 repositories with over 2.5 million packages.

Repology, which is implemented in Python, is maintained on GitHub [2]. Besides the archives of Linux distributions, Repology also integrates some BSD projects' data. In addition, Repology tracks the activities in repositories such as Deb Multimedia, RPM Fusion, Chocolatey, F-Droid, Homebrew, and many others.

In short, Repology acts as a search engine for packages in repositories of Linux distributions and beyond. Since it offers numerous evaluation options, it provides users with many insights and comparisons of individual packages. In this way, you can discover, for example, which distribution is best suited to your needs in terms of cutting edge software, the number of packages provided, or other criteria.

As an introduction, I will use Repology to quickly find out which distributions provide the latest version of Firefox. As background, web applications always need to be up-to-date because of constant new attack scenarios. Ideally, only one or two days should pass before a new or patched version of Firefox or any other web application becomes available in the package sources.

Useful Data

Repology's data is of interest to developers, package maintainers, and end users alike. While developers can discover which distributions offer their software, package maintainers can keep track of new versions of the applications they package and see at a glance which maintainers are packaging the same software for other distributions.

End users know when new software versions become available and can jog package maintainers' memory at their distribution of choice if needed. Apart from this, browsing in Repology is an extremely entertaining pastime.

Fox Hunt

Before getting started, you need to know that Repology refers to everything that has to do with a software application (Firefox in our case) as a project. When you open the web app, you'll find six tabs at the top of the page: *Projects*, *Maintainers*, *Repositories*, and *Tools* on the left and *News* and *Docs* on the right.

I'll start our search in the *Projects* tab where I'll enter *firefox* in the search bar. Clicking on *Advanced* lets you refine the search, for example, by searching in special repositories for the package or for Firefox packages by a specific maintainer. I'll just do a simple search and select Firefox (*firefox**) from the displayed projects (Figure 1).

To the right of the project name in Figure 1, the *Spr* column (which

Lead Image © Oleksiy Mark, 123RF.com
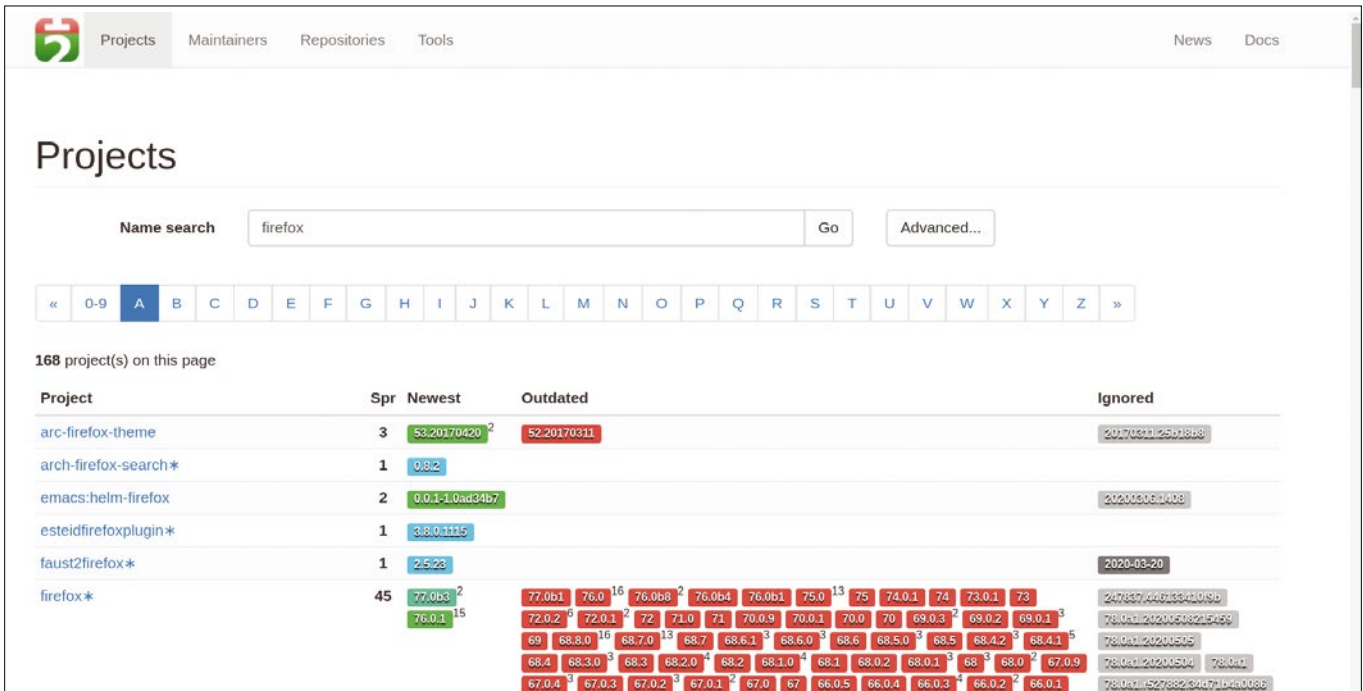
Figure 1: Repology views Firefox as a project, which includes all packages that have Firefox in their name. We are interested in the *firefox** entry at the bottom, which stands for the web browser.

stands for spread) indicates the number of repository families in which the project is found. For example, Debian Stable, Testing, Unstable, Oldstable, and Experimental count as one family. Clicking on *firefox** will open a view of the repositories that provide Firefox (Figure 2).

Many Foxes

In the upper left corner, you can see how many Firefox packages Repology recognizes. These include Firefox, Firefox ESR, Firefox Developer, and all the other variants in all represented versions. Under the *Category* section, you can see where the package is categorized in a

distribution. On the far right below *Maintainer(s)*, you will find the package maintainer(s). Scroll down to the bottom of the page (a long way for the Firefox list, admittedly) to find the archives that currently do not offer Firefox packages.

Green Is the Color

In order to be able to assess whether information is up to date, you need to pay attention to the color codes. A chart explaining what each color means can be found at the bottom of each page or in the documentation [3]. If you are always looking for the latest versions, you want to watch out for the color green.

In Figure 1, Firefox 76.0.1 is green, but Firefox 76.0, which was released only four days earlier at the time of writing, is highlighted in red and therefore considered deprecated. The newest version is beta version 77.0b3, shown in turquoise.

Sometimes an outdated version will not appear in red. Instead, it will have a yellow *legacy* label, which means that this version is still available for compatibility reasons (Figure 3).

Deeper and Deeper

In addition to the versions currently available through the distributions, you can also select *Packages*, *Information*, *History*, *CVEs*, *Related*, *Badges*, and *Report* in the buttonbar at the top. Each

Figure 2: The different Debian branches offer different versions of the web browser. Versions marked in red are outdated; Yellow indicates a legacy version that is still maintained for compatibility reasons.

Figure 3: Using Python as an example, it is easy to see that the obsolete Python 2 version will be with us for a while longer for compatibility reasons in some distributions, while other distributions have already radically removed it.

provides a different view of the information for a given package.

Clicking on *Packages* opens up a wealth of additional information on the various Firefox packages per version and distribution, allowing comparisons between the packages of different distributions (Figure 4).

If there is an official maintainer for a package, clicking on their email address reveals the other distributions for which they maintain this package. *Package related links* takes you to more information, such as the GitHub site, package information on dependencies and licenses, or which maintainer signed the package.

Unlike the *Versions* and *Packages* buttons, which focus on the package, *Information* lists data from all packages and groups it by type (e.g., versions of all known packages, summaries of all known packages, or links for all known packages).

Who's Been Busy?

The still experimental *History* button chronologically lists all actions related to Firefox (Figure 5). This view, like some others, can be set to *Enable autorefresh*, which causes the page to automatically refresh the content.

Related lists connections from Firefox to other projects. The tool identifies related projects by recursively matching URLs on the package homepage.

Badges includes code snippets used to create badges that summon up information from Repology to be included on other websites as a badge. Badges come in different formats, determined by the code snippets.

Figure 4: Clicking on *Packages* provides detailed information about the Firefox versions in a distribution. This ranges from information about the website and package maintainer to links to the build script.

Figure 5: *History* **provides chronological information with hourly updates on changes to the package under review across all monitored repositories.**

Report lets you report problems in regards to how a project is displayed. Errors include the wrong latest version number, an unrelated entry that needs to be split into a new project, or entries for the same project with different names that need to be merged into a single project. Errors are virtually unavoidable with the abundance of cross-referenced information.

Who Maintains Firefox?

To find out who bundles Firefox for which distribution, click on *Maintainers* in the upper left corner. For a successful search, you need the maintainer's email address or nickname, which you should find in the *Versions* view with any luck.

I wanted to see what Mike Hommey, Mozilla developer and Debian package maintainer for Firefox, has been doing. After entering his Debian email address,

Figure 6: **The Maintainer view shows where packages by Firefox maintainer Mike Hommey are used.**

Figure 7: *Updates* shows when activity in each repository last took place. Among other things, both successful and failed build processes are displayed.

clicking on the link opens the page showing Hommey's activities in terms of Firefox packaging. This includes information about distributions that use these Firefox packages, the number of packages, and how up-to-date they are. A click on the respective distributions to which Hommey delivers takes you directly to the distribution page within Repology with further general information (Figure 6).

Statistics and Graphs

The *Repositories* tab leaves the Firefox project and gives a general overview of Repology divided into the subcategories *Statistics*, *Packages*, *Graphs*, *Updates*, and *Fields*.

Statistics displays information about the number of packages, how up-to-date they are, and the number of maintainers, broken down by distribution and their last releases. At the time of publication, Repology is aware of 190,374 projects with a total of 2,507,760 individual packages.

Packages provides details about packages in each distribution. The abbreviations above the columns are defined as soon as you mouse over them.

Graphs evaluates the information in graphs, while *Updates* shows when a distribution last updated the packages (Figure 7). In *Fields*, Repology reveals what information the distributions provide for their packages. This ranges from a summary of the software and its maintainer to the website and license.

Additional Information

The *Tools* tab provides additional tools. *Project by package* lets you generate a link to Repology project pages and badges. *Trending and declining projects* shows projects that have been added or deleted in the past month. This tool is practical if you are looking for interesting projects that are just starting to take off.

Conclusions

If you are interested in more than just using Linux and want to look beyond

the scope of your distribution of choice, Repology offers a cornucopia of current package information. All of this information is interlinked, making it easy to search or just to have a quick look at the huge database. For more information on how Repology collects this data, check out Repology's ruleset on GitHub [4]. ▪▪▪

Info

[1] Repology:
   https://repology.org

[2] Repology on GitHub:
   https://github.com/repology

[3] Color codes:
   https://repology.org/about

[4] Repology ruleset:
   https://github.com/repology/repology-rules

Author

Ferdinand Thommes lives and works as a Linux developer, freelance writer, and tour guide in Berlin.

Public Money

Public Code

Modernising Public Infrastructure
with Free Software

MakerSpace

Steampunk portable Rasp Pi lunchbox
Moveable Feast

A lunchbox-style portable Raspberry Pi computer provides external control for a steampunk robotic skull. *By Rob Reilly*

A s "Dr. Torq," I play a moderately eccentric 1880s-era inventor who develops physical computing projects with modern off-the-shelf parts; then I share lessons learned through how-to articles and conference tech talks. The steampunk aesthetic inevitably generates a lot of curiosity, so it naturally figures prominently in many of my designs. One of my designs, a robotic skull, needed additional controls, so I decided to create a steampunk lunchbox-style portable computer.

Why a Steampunk Lunchbox Computer?

Hedley, my steampunk robotic skull (Figure 1), originally had an onboard Raspberry Pi brain. The Rasp Pi sent speech data from a processing program to an Arduino, actuating the jaw. It also had a JeVois smart machine vision sensor, and I used a webcam program like `luvcview` [1] to monitor whatever the sensor recognized. I built a little outboard monitor to watch the vision sensor feed, effectively turning Hedley into a stealth desktop computer. Hedley even appeared with me at a couple of conferences and ran my LibreOffice session slides. Although Hedley worked, it always felt kind of clunky.

This led to the idea of using a separate portable Linux computer to direct Hedley's actions while also supporting conference tech talk activities. A lunchbox layout style would allow ample freedom for over-the-top design elements and give a cool "open frame" steampunk look. Here, I look at the lunchbox machine in its present form, the software, and how it all came into "sitting on

Figure 1: Hedley, the steampunk robotic skull.
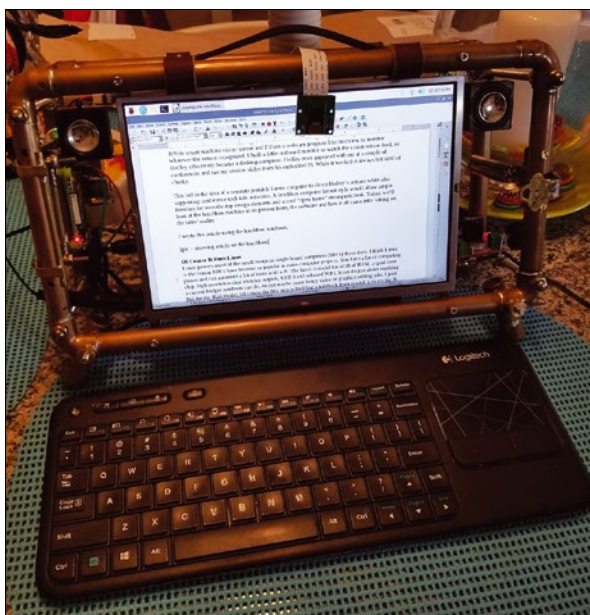
Figure 2: The steampunk Linux lunchbox computer showing this article in LibreOffice Writer.

the table" reality. In fact, this article was written with the lunchbox notebook (Figure 2).

Of Course It Runs Linux

Linux powers most of the small-footprint single-board computers (SBCs) these days, and I think Linux is the reason they have become so popular in nano-computer projects. With the considerable computing power of a Raspberry Pi, you can automate many jobs. The latest model 4 has 4GB of RAM, a quad-core chip, high-resolution dual-monitor outputs, USB 3, and onboard WiFi. (See Table 1 for the hardware components.) It can do just about anything a current budget notebook can do, except maybe super-heavy video or graphics editing. I paid $55 for the 4GB model. Of course, the first step in building a notebook from scratch is to get the Rasp Pi module up and running with a fresh Linux build.

Unlike the old days, Linux system installation on a Pi is very straightforward. Basically, I performed a stock Raspberry Pi OS build. First, I downloaded the latest Raspberry Pi OS [2] (formerly Raspbian Linux). At the time of the project, this was `2020-05-27-raspio-buster-armhf.zip`. Next, I unzipped the package on my ASUS Linux notebook daily driver and used the `dd` command to make short work of writing the ISO image to a Samsung EVO+ 32GB microSD card, which I then popped into the Rasp Pi before powering up the board. Lastly, I ran through the *resize the main partition* menu and set up wireless access with my local router SSID. I enabled SSH and changed the memory split to 256MB for optimal video performance. SSH is useful for copying code and working files from my ASUS notebook to the lunchbox Pi.

The Tube Frame

With the Rasp Pi operating system squared away, I will look at the physical hardware side next, but before I get into the details of the tube frame, let me explain why I chose a lunchbox layout.

Figure 3: Lunchbox with the keyboard cover latched on the front.

Table 1: Parts List

Part	Source
Sourced Online	
JeVois smart vision camera:	http://jevois.org/
Raspberry Pi 4 (4GB model)	https://www.adafruit.com/product/4296
Logitech K400r wireless keyboard/mousepad	https://www.logitech.com/en-us/product/wireless-touch-keyboard-k400r
Samsung EVO+ 32GB micro-SD card	https://www.bestbuy.com/site/samsung-evo-plus-32gb-microsdhc-uhs-i-memory-card/5785401.p
10.1-inch color LCD screen with HDMI driver board and controls	https://www.banggood.com/10_1-Inch-1280x800-HD-Display-TFT-LCD-Module-Kit-For-Raspberry-Pi-p-1109750.html
10W per channel audio amplifier board	https://www.banggood.com/12V-Mini-Hi-Fi-PAM8610-2X10W-Audio-Stereo-Amplifier-Board-Dual-Channel-p-933675.html
15,000mAh super polymer lithium ion battery	https://www.banggood.com/DC-12V-15000mAh-Super-Rechargeable-Protable-Lithium-ion-Battery-Pack-p-970196.html
5MP Raspberry Pi camera with 12-inch flat cable	https://www.amazon.com/Raspberry-Camera-Module-Megapixels-Sensor/dp/B07L82XBNM/ref=sxin_7_ac_d_pm
HDMI to micro-HDMI adapter	https://www.banggood.com/1_5M-Micro-HDMI-Male-D-Type-to-HDMI-Female-A-Type-Adapter-Support-3D-HD-1080P-V1_4-For-FPV-Goggles-p-1098351.html
Micro-HDMI to HDMI adapter	https://www.banggood.com/Micro-HDMI-D-Type-Male-to-HDMI-A-Type-Female-Adapter-Connector-Cable-p-1143811.html
12V to xV regulator (x2)	https://www.amazon.com/Regulator-DROK-Converter-Step-Down-Transformer/dp/B0758ZTS61/ref=sr_1_12
Locally Sourced	
Short audio aux cable	
12V, 1A wall wart	
Miniature 8-ohm speaker (x2)	
Latch spring (x2)	
Various brass and steel #8 machine screws, nuts, and washers	
1/4-inch threaded rod, 5 inches long (x2)	
1/2-inch copper tubing	
1/2-inch copper elbows	
3/16-inch brass tubing	
1/4x1/8-inch flat brass	

Figure 4: Frame copper elbow corner detail.

I like a full-sized keyboard and wanted to use the very common Logitech K400r wireless keyboard/mousepad. The dimensions are 14x5.5x1 inches (WxHxD). The keyboard size pretty much drove the frame design because I also wanted to use it as a front cover for the 10.1-inch color LCD screen during transport (Figure 3) or when the machine was not in use.

The tube frame provides a rugged, yet fairly light foundation for mounting the rest of the hardware. It consists of 1/2-inch straight copper tubing sourced from a big-box home improvement store. The tubing comes in 8-foot lengths, so I cut it with a standard plumber's tubing cutter. Corners are 1/2-inch copper elbows. The front and rear rectangular frames are separated by 1/4-inch threaded rods. Copper is pretty easy to solder and machine. I used a regular propane torch and rosin-core solder to fix the tubing to the elbows (Figure 4). A drill press aided in placing accurate holes for the various mounting points in the tubing and elbows. The case ended up measuring 14.5x8x5 inches (WxHxD).

The Rasp Pi configuration of video output and power ports at the bottom and the USB ports at the back established the depth of the lunchbox. To make the lunchbox thinner, I could have turned the Pi so the USB ports were facing the left-hand side of the case. I might try that on the next major iteration.

Brackets, Brackets Everywhere

Brass goes hand-in-hand with copper when building steampunk gadgets. I used 3/16-inch thin-wall brass tubing, along with 1/8-inch-thick by 1/4-inch-wide flat brass stock to build the mounting brackets for the Rasp Pi, LCD screen, video driver board, speakers, audio amp,

and battery. My trusty antique 100/140-watt Weller soldering gun worked well for bracket construction.

Brass is very easy to solder with rosin-core solder and is a key reason I like using it in steampunk devices. You can cut the brass tubing with the same plumber's tool used for the copper tubing. The flat brass stock is easily cut with either straight aviation tin snips or a Dremel with a cut-off disk. I used a sanding disk on the Dremel and a sanding sponge for edge smoothing.

I also fabricated a spring-operated latch on the front to hold the keyboard in place (Figure 5). A fixed bracket protrudes from the left front of the case to hold that side of the keyboard. A couple of speakers are mounted facing toward the front and slightly behind and on either side of the LCD screen. The brackets are attached to the LCD frame. Interestingly, the open space frame design gives a pleasant 3D effect to the sound from the speakers.

Right behind the LCD screen is the video driver board mounted on its own bracket (Figure 6). The video board runs on a direct 12V line to the power board; it has HDMI, VGA, and composite inputs, and it uses a short HDMI to micro-HDMI cable to hook the board to the Raspberry Pi. A second, very short micro-HDMI to HDMI adapter cable connects to a longer cable for any optional external monitors.

Talk About Power

The lunchbox can operate in two different power modes. Most of the time it's hooked up to a 12V, 1.0A generic wall wart repurposed from an old network router with roughly 12W of power capacity. Measuring the consumption with a Kill-A-Watt meter showed the machine used about 11W in normal desktop operation. It idled back to 5W when the screen was blanked.

An on-board 15,000mAh super polymer lithium ion battery connected to the main power supply board sits in a bracket at the bottom of the lunchbox frame (Figure 7) and is charged through the 12V wall wart when the Rasp Pi isn't running. A larger capacity wall wart could be used if I needed to charge the battery while also running the Pi.

The power supply board has a pair of 12V to xV (up to 3A output) regulators

on a small piece of phenolic circuit board. Outputs are selectable by jumpers for from 1.8 to 12V. A female socket on one end connects to the battery. I used a common + /- input power bus setup for ease of future expansion. One regulator supplies a dedicated 5V to the Raspberry Pi. The other regulator is for the steady 12V needed by the audio amplifier board. The color LCD driver board connects to the 12V power directly from the wall wart or battery input power bus. The speakers are powered by a 10W per channel stereo amplifier board with a standard aux cable running to the Raspberry Pi analog audio port (Figure 8).

I plan on adding a switchboard to control all the hardware modules individually. For example, when booting everything at once on the battery, the audio board doesn't initialize correctly because of high current demands. Switching it on after the Pi boots up eliminates that problem. Plus, if I'm not going to listen to audio, the amplifier can be turned off to conserve battery power.

Software

The first piece of software I always load onto my Linux machines is the Synaptic package manager [3] for installing software; `apt` on the command line works well, too, particularly for regular software updates.

As mentioned earlier, I use the `luvcview` tool [4] for viewing the video feed from the Hedley's JeVois smart machine vision sensor, which uses neural network algorithms to recognize 1,000 different objects; it superimposes a box around objects at 30 frames a second and provides a real-time data stream of recognized object

Figure 5: A spring-loaded latch holds the keyboard cover in place.
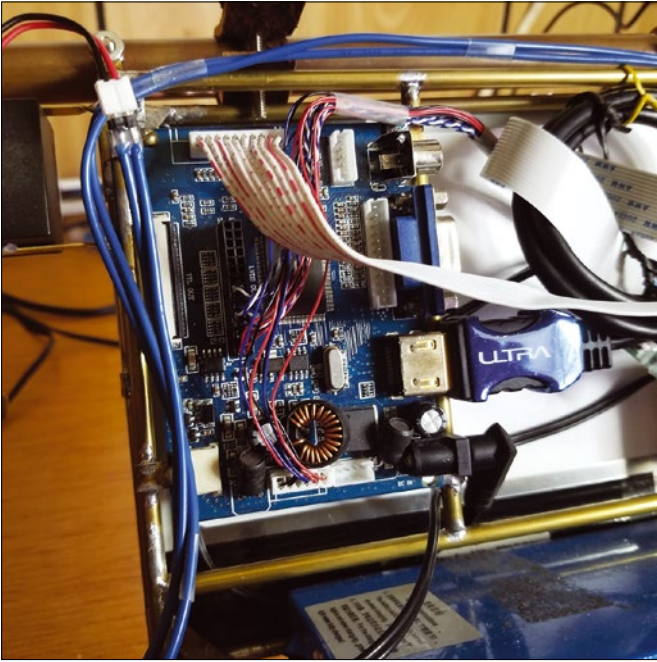
Figure 6: The video board and bracket behind the LCD.
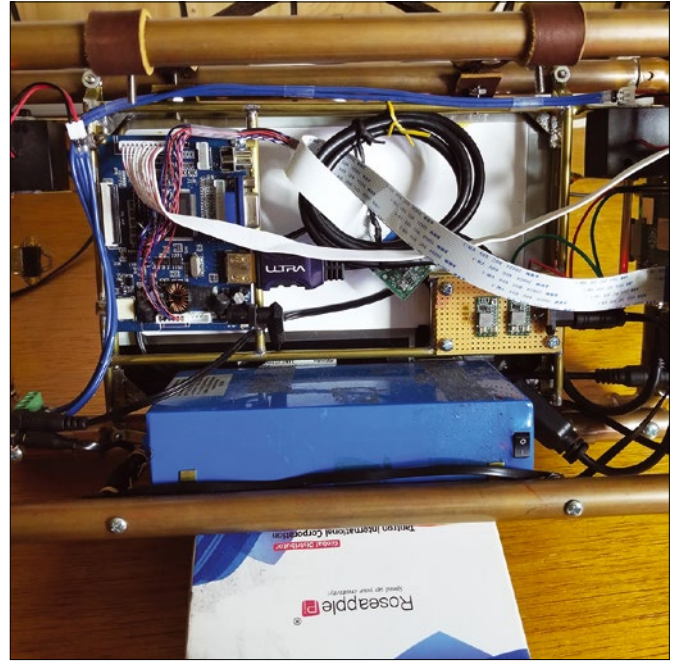
Figure 7: The battery and internal brackets.

names and XY coordinates, so you can track faces or things and perhaps move servos on your robot. I've found `luvcview` to be much more reliable than `guvcview` or other camera applications.

LibreOffice handles all the office-type jobs you'll ever have on a notebook. I mostly use the Writer word processor and Impress presentation manager. I wrote this entire story with LibreOffice on the lunchbox. The only thing I don't like is that, while touch typing, the mousepad sometimes picks up my trailing right little finger movement when pressing the Delete key. That usually moves the cursor somewhere unexpected, and I end up mistakenly deleting a word or paragraph. I'll have to train myself to cross the left-hand index finger over for deletes.

Overall, I can load just about anything on the lunchbox and it will function properly. It even runs exotic programs like `gqrx` for software-defined radio listening.

Future Upgrades

Having a "one-button" shutdown would be nice, which just requires writing a little Python program that runs at startup and watches for a button push. When it gets one, the program makes an operating system call to execute the `sudo shutdown -h now` command. This technique would save having to hunt around on the screen for the shutdown icon and menu.

Another upgrade might be to motorize the wired Raspberry Pi camera with pan and tilt capability. The camera could rise up out of the lunchbox with a lifting mechanism and then use pan and tilt servos to direct the field of view. OpenCV [5] might make it possible to recognize and track faces and do other cool things.

Lastly, I want to develop some type of bus or network system to manage Hedley and Barkley (a soon-to-be-built skeletal robotic companion) for tech talks and shows.

Wrap-up

The lunchbox computer has proven reasonably durable. I've used it off and on for about a month without problems. A

Figure 8: The audio amplifier board and bracket.

fully charged battery runs for about six hours. Bootup is pretty quick from power-on to a working desktop in about 30 seconds.

The steampunk look certainly draws a lot of attention, although one lady at a coffee shop asked if it had anything to do with launching rockets for spaceflight. ∎∎∎

Author

Rob "drtorq" Reilly is an independent consultant, writer, and speaker specializing in Linux/OSS, physical computing, hardware hacking, tech media, and the DIY/Maker movement. He provides a variety of engineering, business, and special project services to individual clients and companies. As a long-time veteran of the tech media, Dr. Torq has posted hundreds of feature-length articles for top-tier tech media and print outlets. He's also presented tech talks at OSCON and other industry venues. Contact him at *doc@drtorq.com* or *407-718-3274*.

Info

[1] luvcview:
https://github.com/ksv1986/luvcview

[2] Raspberry Pi OS:
https://www.raspberrypi.org/downloads/raspberry-pi-os/

[3] Synaptic package manager:
http://www.nongnu.org/synaptic/

[4] luvcview man page:
http://manpages.ubuntu.com/manpages/trusty/man1/luvcview.1.html

[5] OpenCV: *https://opencv.org*

MakerSpace

Halloween candy vending machine
Halloween Machine

A Halloween vending machine frightens visitors, but the braver ones receive a chocolate reward. *By Martin Mohr*

With a project as complex as a Halloween vending machine, you need to take everything step by step; otherwise, you quickly get confused. To begin, you must clarify what purpose the vending machine serves. The young visitors are supposed to start it themselves; it then performs a scary show and hands out a piece of chocolate. The show includes sound and light effects – which doesn't sound too complicated – at first.

Table 1: PiXtend V2 -S- Specs

Digital Inputs and Outputs
8 Inputs (3.3/5/12/24V)
4 Outputs (max. 30V, 0.5A each)
4 PWM/servo outputs (2x16 bits, 2x8 bits)
Analog Inputs and Outputs
2 Voltage inputs (0-5V/0-10V)
2 Voltage inputs (0-10V)
Interfaces
4 GPIOs (5V)
Serial interface (RS-232)
4 Relays (max. 230V, 6A)
Support for up to four DHT11/DHT22/AM2302 sensors (temperature and humidity)
Slot for 433MHz transmitter (radio socket, not included)
Onboard Voltage Regulator
Inputs 12-24V DC (max. 30V)
Output 5V DC/2.4A (supplies PiXtend V2 -S-, Raspberry Pi, and connected USB devices)
Miscellaneous
Real-time clock (battery buffered)
Retain/persistent memory, 32-byte flash EEPROM
Designed for Raspberry Pi models B+/2B/3B
Certifications: CE, RoHS

Chassis

A stable wooden beam forms the basis of the machine. All of the components can be attached to it easily. The temptation to use warped beams just because they are available and cost almost nothing will probably cause more trouble than spending money on good material. The specific machine from this example has a height of around 190cm (~ 6ft) with a footprint of 60x60cm (~ 2x2ft). However, no precise size and material specifications are made at this point, just a few tips.

Above all, you need to make sure that the body does not easily tilt or tip over. For this purpose, it is useful to add weights to the base (e.g., heavy stones). Wheels to roll the machine are useful, too. If you construct the vending machine in your garage or basement, test beforehand whether it will fit through the doors that deliver it to its final destination. I have had to disassemble and shorten a completely finished construction in the past just to get it outside.

Brain Power

Every machine needs a control system. I have used the PiXtend V2 -S- (Table 1) a few times in Halloween projects, and it has always performed excellently [1]; it has all the I/O interfaces I need and is easy to program. This time I chose to use the Python scripting language because it has useful libraries for all the components I use.

Figure 1: Conveyor belts with chocolate.

Hardware Setup

The construction of the vending machine is divided into two areas: the conveyor belts to output the chocolate and the movement of the eyes and mouth. Figure 1 shows the chocolate store, which consists of two conveyor belts that alternately start up and hand out a piece of chocolate.

The sensors are capacitive proximity switches (see the "Zero-Contact Switch" box) with negative switching logic: As soon as something is in front of the sensor, the PiXtend input signal drops to 0V. If nothing is in front of the sensor, the voltage is 24V.

The conveyor belt starts as soon as the creepy show starts, so the chocolate is

Zero-Contact Switch

In modern installations, zero-contact switches are increasingly found alongside traditional mechanical switches. The three most common sensors are:

• A reflex light barrier, which is based on the principle of light reflection and comprises an infrared transmitter and an IR receiver. As soon as an object reflects enough light, the photoelectric switch is triggered. The range can be several meters.

• An inductive proximity switch, which creates a magnetic field and triggers as soon as a metallic object interferes with the field. The range is often limited to a few centimeters.

• A capacitive proximity switch, which generates an electric field and is triggered as soon as this field is disturbed. It has a similar range as the inductive sensor; however, the electric field can also be disturbed by non-metallic objects.

ready in time for the output. When the input to which the initiator is connected experiences a positive edge (transition from 0 to 24V) the conveyor belt stops. The belt motors operate at 12V, which they receive from a separate power supply. To get an idea of how the conveyor belt works, watch the video on YouTube [2].

The eyes and the mouth are made of WS2812 LED panels, which a Raspberry Pi controls. It makes sense to order the panels directly from China [3], because they are quite expensive on Amazon. To work with the panels later, you need additional information about their design. The LEDs, soldered in meanders (loops), cannot be addressed directly by physical position. The program contains a matrix that supports easier access to the individual LEDs.

A quick look at the circuit diagram (Figure 2) shows how the sensors and motors for the conveyor belt and the driver for the LEDs work together. The LEDs require 5V, but the GPIO on the Rasp Pi only supplies 3.3V. You therefore need to add a driver component between the Rasp Pi and the WS2812 to increase the voltage: I used an SN74HCT125 TTL [4] (hereafter 74125).

Note that the device needs a power supply, which can be provided by the GPIO on the Rasp Pi.

The WS2812 programmable diodes can handle 16,777,216 different colors. They consist internally of three individual LEDs for red, green, and blue. Each of the three colors can be controlled by 1 byte, so you can mix the three colors as required. The 3 bytes reach the LED from a shift register, and the protocol is very time-critical. A high-level language such as Python is therefore not really suitable in this case, but there is a Python library that uses assembler code to transfer the data.

The use of a shift register makes it possible to control almost any number of these LEDs as a long chain. On the Rasp Pi, you only need one GPIO pin to control each LED, no matter how many LEDs are in the chain; however, not every pin is suitable for this usage, because it needs certain timing characteristics. More detailed information can be found on the homepage of the library project [5].

At maximum brightness, an LED can consume up to 60mA, which you should keep in mind when dimensioning the power supply. If you do not turn the brightness up that high, the 5V output of the PiXtend (100mA) is fine for supplying the devices with power. If this is not enough, you need to connect both 5V outputs in parallel to reach 200mA.

The PiXtend is connected to the LEDs with a special flat ribbon cable (Fig-
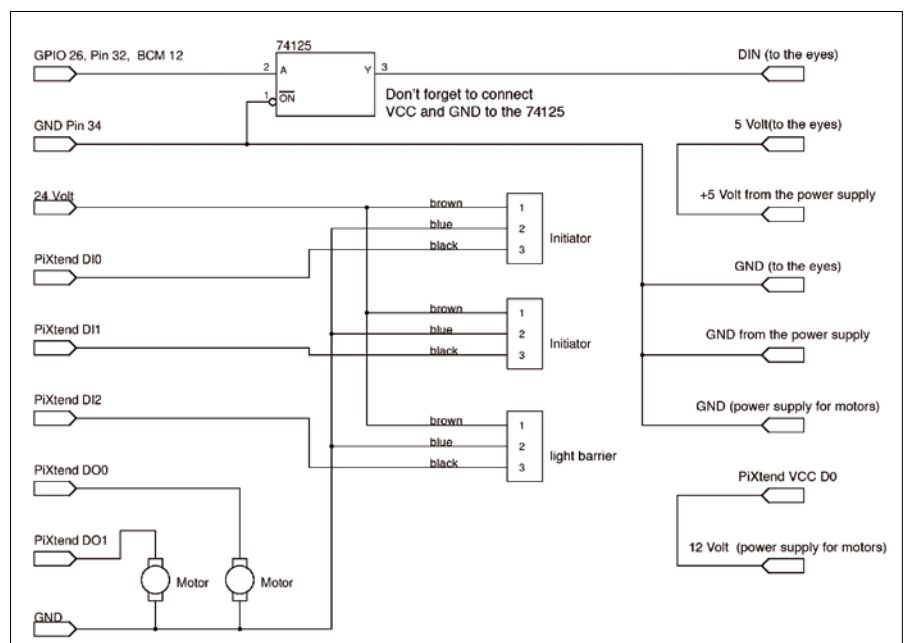
Figure 2: Circuit diagram for the Halloween machine.
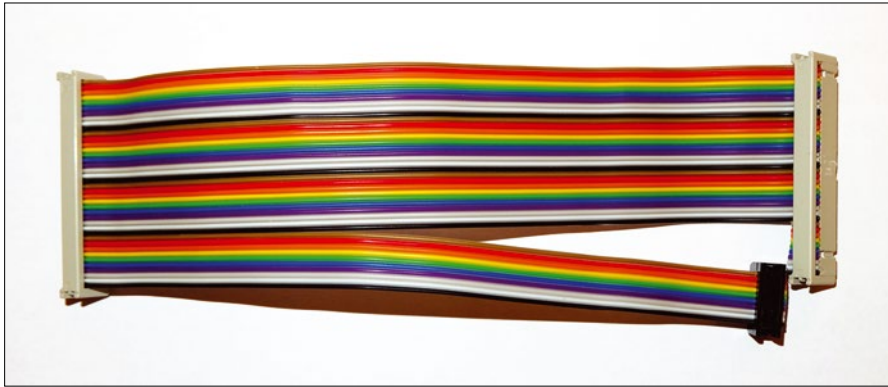
Figure 3: The flat ribbon cable that connects the WS2812 LEDs to the PiXtend.

Listing 1: Installing PPL

```
01 $ cd ~
02 $ sudo apt update
03 $ sudo apt upgrade
04 $ sudo apt install build-essential python-dev python-pip unzip wget scons swig
05 $ wget https://github.com/jgarff/rpi_ws281x/archive/master.zip  unzip master.zip
06 $ cd rpi_ws281x-master
07 $ sudo scons
08 $ sudo pip install rpi_ws281x
09 $ cd ~/rpi_ws281x-master/python
10 $ sudo PYTHONPATH=".:build/lib.linux-armv7l-2.7" python examples/strandtest.py
```

ure 3), which routes out GPIO header pin 32. If necessary, you can easily make this yourself.

Software

The preconfigured Python image from the PiXtend website [6] lets you start programming as quickly as possible. Make sure the image matches the PiX-tend model you use. To do so, search for the *PiXtend Python Library* heading on the page for the respective device.

Once you have downloaded the image and written it to an SD card, the PiX-tend Python Library (PPL) is available directly. You only need to install the li-brary to control the WS2812 LEDs. These steps are summarized in the first eight lines of Listing 1. To test whether the library installation was successful, run the strandtest.py program (lines 9 and 10).

In the source code, you can configure different settings for the LEDs, including the LED_BRIGHTNESS parameter for the brightness of the diodes. You will want to choose a value that is as low as possible, because the LEDs consume a lot of cur-

Listing 2: ws.py

```
01 import time
02 from neopixel import *
03 import argparse
04
05 LED_COUNT      = 384
06 LED_PIN        = 12
07 LED_FREQ_HZ    = 800000
08 LED_DMA        = 10
09 LED_BRIGHTNESS = 16
10 LED_INVERT     = False
11 LED_CHANNEL    = 0
12
13 TRANSLATE = [0, 15, 16, 31, 32, 47,
    48, 63, 64, 79, 80, 95, 96, 111,
    112, 127, 128, 143, 144, 159, 160,
    175, 176, 191, 192, 207, 208, 223,
    224, 239, 240, 255, 256, 271, 272,
    287, 288, 303, 304, 319, 320, 335,
    336, 351, 352, 367, 368, 383, 1, 14,
    17, 30, 33, 46, 49, 62, 65, 78, 81,
    94, 97, 110, 113, 126, 129, 142,
    145, 158, 161, 174, 177, 190, 193,
    206, 209, 222, 225, 238, 241, 254,
    257, 270, 273, 286, 289, 302, 305,
    318, 321, 334, 337, 350, 353, 366,
    369, 382, 2, 13, 18, 29, 34, 45, 50,
    61, 66, 77, 82, 93, 98, 109, 114,
    125, 130, 141, 146, 157, 162, 173,
    178, 189, 194, 205, 210, 221, 226,
    237, 242, 253, 258, 269, 274, 285,
    290, 301, 306, 317, 322, 333, 338,
    349, 354, 365, 370, 381, 3, 12, 19,
    28, 35, 44, 51, 60, 67, 76, 83, 92,
    99, 108, 115, 124, 131, 140, 147,
    156, 163, 172, 179, 188, 195, 204,
    211, 220, 227, 236, 243, 252, 259,
    268, 275, 284, 291, 300, 307, 316,
    323, 332, 339, 348, 355, 364, 371,
    380, 4, 11, 20, 27, 36, 43, 52, 59,
    68, 75, 84, 91, 100, 107, 116, 123,
    132, 139, 148, 155, 164, 171, 180,
    187, 196, 203, 212, 219, 228, 235,
    244, 251, 260, 267, 276, 283, 292,
    299, 308, 315, 324, 331, 340, 347,
    356, 363, 372, 379, 5, 10, 21, 26,
    37, 42, 53, 58, 69, 74, 85, 90, 101,
    106, 117, 122, 133, 138, 149, 154,
    165, 170, 181, 186, 197, 202, 213,
    218, 229, 234, 245, 250, 261, 266,
    277, 282, 293, 298, 309, 314, 325,
    330, 341, 346, 357, 362, 373, 378,
    6, 9, 22, 25, 38, 41, 54, 57, 70,
    73, 86, 89, 102, 105, 118, 121, 134,
    137, 150, 153, 166, 169, 182, 185,
    198, 201, 214, 217, 230, 233, 246,
    249, 262, 265, 278, 281, 294, 297,
    310, 313, 326, 329, 342, 345, 358,
    361, 374, 377, 7, 8, 23, 24, 39, 40,
    55, 56, 71, 72, 87, 88, 103, 104,
    119, 120, 135, 136, 151, 152, 167,
    168, 183, 184, 199, 200, 215, 216,
    231, 232, 247, 248, 263, 264, 279,
    280, 295, 296, 311, 312, 327, 328,
    343, 344, 359, 360, 375, 376]
14
15 strip = Adafruit_NeoPixel
    (LED_COUNT, LED_PIN, LED_FREQ_HZ,
     LED_DMA, LED_INVERT, LED_
     BRIGHTNESS, LED_CHANNEL)
16 strip.begin()
17
18 f = open("data.bin","rt")
19 data=f.readlines()
20 bitcounter=0
21 image=2
22 for line in data:
23   if len(line)<3:
24     bitcounter=0
25     image=image+1
26     strip.show()
27     time.sleep(0.1)
28   if not(line.find("#")):
29     continue
30   bits = line.split(" ")
31   for bit in bits:
32     bit=bit.strip()
33     if len(bit) > 0:
34       if bit=='1':
35         strip.setPixelColor
           (TRANSLATE[bitcounter],
            Color(0,255,0))
36       else:
37         strip.setPixelColor
           (TRANSLATE[bitcounter],
            Color(0,0,0))
38       bitcounter=bitcounter+1
39 f.close
```

Listing 3: data.bin (Excerpt)

```
# 2 3 4 5 6 7 8 1 2 3 4 5 6 7 8 1 2 3 4 5 6 7 8 1 2 3 4 5 6 7 8 1 2 3 4 5 6 7 8 1 2 3 4 5 6 7 8
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0
0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0
1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0
1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0

# 2 3 4 5 6 7 8 1 2 3 4 5 6 7 8 1 2 3 4 5 6 7 8 1 2 3 4 5 6 7 8 1 2 3 4 5 6 7 8 1 2 3 4 5 6 7 8
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0
0 1 1 1 1 1 1 0 0 1 1 1 1 1 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0
1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0
1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0
0 1 1 1 1 1 1 0 0 1 1 1 1 1 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0
```


Figure 4: The vending machine's eyes and mouth.

rent at full light intensity, and they also heat up.

Control

The `ws.py` program (Listing 2) [7] controls the eyes and mouth. At the start of the program you will find the `TRANSLATE` table that enables linear access to the individual LEDs. As mentioned earlier, the way the LEDs are installed on the panels is a little strange.

The information as to which LED is switched is provided in the `data.bin` file; Listing 3 shows an excerpt. Bit patterns control the LEDs. You can store any number of patterns for `ws.py` to send to the LEDs in sequence. It parses the file line-by-line and converts the data to commands for the LEDs. An empty line initiates a new bit pattern; the program skips comment lines that start with a hash symbol (`#`).

Program

The control program (Listing 4) continuously queries the PiXtend's inputs and starts the conveyor belts alternately when the light barrier is triggered. In parallel to starting the belts, it calls two additional programs: One controls the LEDs, and the other plays back the matching creepy sound. If the `play` command doesn't work as intended, you can try connecting and configuring an external sound card to the

Outtakes – What Went Wrong

As early as the planning and construction phase, the Halloween vending machine proved to be a tough cookie. In the first version, the sweets were prone to fall out of the magazine that was designed to deliver them. After a few weeks of trial and error, I abandoned this idea: it proved impossible to even halfway control the number of sweets handed out. Either none came out, or a whole load were handed out.

The arrangement of the LED panels for the eyes and mouth almost drove me crazy. When I was building the machine in the garage, I burned my fingers on the soldering iron – this hadn't happened to me in ages. Hundreds of screws fell during the assembly work and predictably always landed in the most inaccessible corner of the garage.

The first draft of the software was developed for a PiXtend 1.3, but the machine was initially equipped with a PiXtend 2.0. Because it is not possible to simply swap the SD card from one model to another, I had to change the PiXtend board at the last minute. For unknown reasons, I also had problems with the Rasp Pi's on-board sound. I decided to disable it and use an external sound card instead.

The WS2812 LEDs take their voltage from the 5V output of the PiXtend, which works fine as long as you run the LEDs at low intensity. However, if you turn them up to test the animation in sunlight, quite strange things happen with the LEDs. Originally, the machine had waving arms, but during testing they turned out to be a hazard zone for the children's fingers, so I left them out in the final version.

Finally, some problems cropped up with the machine on Halloween. My camera caused problems in the dark, so I don't have any pictures of that year's Halloween invention in action. To make matters worse, a fog machine set up to support the creepy effect finally caused a massive short circuit – and the ignominious end of that year's Halloween display. I've heard of people's hair turning white from fright, but in my case, the project rapidly caused my hair to turn gray.

Listing 4: halloween.py

```
01 #!/usr/bin/env python
02 # coding=utf-8
03
04 from __future__ import print_function
05 # Import Pixtend class
06 from pixtendlib import Pixtend
07 import time
08 import sys
09 import threading
10 import os, platform
11
12 p = Pixtend()
13
14 # Open SPI bus for communication
15 try:
16   p.open()
17 except IOError as io_err:
18   # Print error text and delete PiXtend instance
19   print("Error opening the SPI bus! Error is: ", io_err)
20   p.close()
21   p = None
22
23 def thread_sound():
24   os.system("play gost.wav")
25
26 def thread_animation():
27   os.system("sudo python ws.py")
28
29 # ---------------------------------------------------
30 # Main Program
31 # ---------------------------------------------------
32 if p is not None:
33   print("Running Main Program - Hit Ctrl + C to exit")
34   # Set some variables needed in the main loop
35   is_config = False
36   switch = True
37   pressed = False
38   while True:
39     try:
40       # Using Auto Mode for optimal SPI bus usage
41       if p.auto_mode() == 0:
42         if not is_config:
43           is_config = True
```

```
44        # Space for setup
45     print (p.digital_input0,":",p.digital_input1)
46     if p.digital_input2 == p.ON and pressed==False:
47       pressed=True;
48       x = threading.Thread(target=thread_sound)
49       x.start()
50       y = threading.Thread(target=thread_animation)
51       y.start()
52       if switch: p.digital_output0 = p.ON
53       else : p.digital_output1 = p.ON
54       switch=not(switch)
55     if p.digital_output0 == p.ON and di0_old == 0 and
         p.digital_input0 == p.ON:
56       p.digital_output0 = p.OFF
57       pressed=False
58     if p.digital_output1 == p.ON and di1_old == 0 and
         p.digital_input1 == p.ON:
59       p.digital_output1 = p.OFF
60       pressed=False
61     di0_old = p.digital_input0
62     di1_old = p.digital_input1
63   else:
64     print("Auto Mode - Communication is not yet up...
             Please wait...")
65   # Wait at minimum 0.1sec or 100ms before getting
       new values
66   time.sleep(0.1)
67 except KeyboardInterrupt:
68   # Keyboard interrupt caught, Ctrl + C, now clean up
       and leave program
69   p.close()
70   p = None
71   break
72 else:
73   # If there was an error when opening the SPI bus
       interface, leave the program.
74   print("")
75   print("There was a problem with the PiXtend
         communication. Quitting.")
76   print("")
```

Rasp Pi. Figure 4 shows the eyes and mouth. To see the whole setup working, check out my YouTube video [8].

Conclusions

I have never known so many things to go wrong as did with this 2019 Halloween project. If you enjoy pratfalls, bad luck, and breakdowns, take a look at the box "Outtakes – What Went Wrong."

Because the little visitors always came in groups, it soon became clear that the capacity of the conveyor belt was not up to the task. Stocking up with more chocolate in the dark caused major problems. Have a flashlight ready or – better still – install a light in the machine.

In spite of all the breakdowns, both big and small visitors enjoyed the vending machine. Some were so enthusiastic that they came by several times during the evening. A handful of children and adults with an affinity for technology even took the opportunity to take a detailed look inside the machine. All in all, the Halloween vending machine was great fun for everyone involved. Version 2.0 is already planned for this year. ∎∎∎

Info

[1] PiXtend:
https://www.pixtend.de/downloads/
pixtend-v2-downloads-english/

[2] Video of conveyor belts:
https://youtu.be/3GgB42z9lZ8

[3] LED panel: https://www.aliexpress.
com/item/32826934452.html

[4] SN74HCT125 TTL:
https://www.ti.com/product/
SN74HCT125

[5] Library for WS2812:
https://github.com/jgarff/rpi_ws281x

[6] PiXtend download page:
https://www.pixtend.de/downloads/

[7] Code for this article:
ftp://ftp.linux-magazine.com/pub/
listings/linux-magazine.com/238/

[8] Movie of eyes and mouth:
https://youtu.be/mvK4GS5FCJM

LINUXVOICE ▶

A few years ago, we couldn't stop talking about the Linux desktop. It seemed that Linux was riding a big, beautiful wave to glory and would soon snatch massive market share from the Windows and macOS desktop systems. Linux is here to stay of course, but those dreams of conquering the desktop didn't work out exactly like the evangelists were planning. What happened? FUDD? Fragmentation? Too many cooks stirring too many pots? Sure all of that, but was there something else? In this month's LinuxVoice, we look at perceptions of Linux among non-users and analyze the state of the Linux desktop through the lens of a human behavior theory known as social proof. Also inside, we show you how to record command-line sessions with asciinema, and we take a look at UltraScreen – an open source screen-sharing tool that works very well across a router.

Image © Olexandr Moroz, 123RF.com

MADDOG'S DOGHOUSE

IoT does require electricity and other resources, but is that cause for concern? BY JON "MADDOG" HALL

Jon "maddog" Hall is an author, educator, computer scientist, and free software pioneer who has been a passionate advocate for Linux since 1994 when he first met Linus Torvalds and facilitated the port of Linux to a 64-bit system. He serves as president of Linux International®.

Considering Computing Resources

Recently I was giving a talk at a virtual conference on how to start your own business using free culture. I mentioned the basic, underlying premise that no one ever buys software or hardware. People buy a solution to a problem, whether that problem is running a business or playing a game. The fact that they find it easier to solve that problem with computer hardware and software is a side issue.

A young man started giving a presentation about how we had gone down the "technological rathole" and how technologies like IoT should be eliminated due to the amount of data transmission that would be used by IoT as well as the amount of electricity that was used by computing in general.

He gave his credentials as being a developer in a networking organization.

I must admit to being shocked, for I view the use of computing resources as something completely different.

Yes, there are people who leave their computing systems on all the time even when they are not actively using them, but these are typically low power devices that use much less electrical power in the long run than the alternative methods of solving the same problem.

There are people who give little or no thought to where and how they should be "solving their problem," choosing to ignore a good solution for "one that works" however poorly.

As people are beginning to realize, not everyone needs a very powerful desktop or laptop computer to do the computing they need day to day. Many people are doing web work or types of "office work" that only really require a Chromebook or Netbook type of computer.

It was slightly different when I started in computing in 1969. A single transistor could cost $1.50 in US dollars, with a power transistor costing $2.50.

Even mainframe computers costing millions of dollars (and that was when *one* million dollars was "a lot of money") typically did not have even a 32-bit address space, much less a 64-bit address space – it was simply too expensive. Today you can get a million transistors in a single chip for less than $1, with most CPUs having billions of transistors on them.

Computers "in the day" used to use kilowatts of electricity and use water cooling to execute solutions that would be deemed "simple" today.

I remember a comparison long ago saying that if computers and automobiles had evolved at the same pace, cars would be able to drive around the world on a cup of gasoline. Of course this was a simplistic comparison, but computers have come a long way.

Yet go into a business and you may find many desktop computers that are turned on, with disks filled with data, using electricity and generating heat that air conditioners have to remove.

What most people want today is fast and consistent access to the data that they need at this moment. People do not want to wait for their computer (or phone) to boot, and they want their data to be sent to wherever they want to be.

The person who was giving this talk was really concerned about how IoT would be sending gigabytes of data over the network and how much electricity would used and how many new routers it would need and so forth. He was particularly anxious about the amount of energy it would use for autonomous cars.

Today the controls of a car are seldom controlled physically. What used to be mechanical linkages that transformed to hydraulic controls are now electronic sensors and actuators. It is cheaper to put in electric windows than it is mechanical hand-crank windows. Most cars have multiple computers in them already.

I pointed out that if cars were truly autonomous we would need a lot fewer cars. I would foresee more cars "rented" only for the time you actually need them, with no cars parked in your driveway or at your place of business. The number of cars manufactured would drop. The number of buildings and parking lots needed for garages would drop dramatically. The amount of resources saved would dwarf the needs for producing the computers (which would probably already exist in the car anyway).

Cars would rarely be coming to a stop at a stop sign or stop light, but simply slowing down as they got to an intersection, and would weave through the intersection. Energy would be saved.

Much of the processing of data would be done "on the edge" of the local network, never having the raw data sent back to a central processing place.

While we should pay attention to data and electricity costs, time has shown that use of computers and data processing typically saves energy and resources with things getting better, not worse. ∎∎∎

Social proof and the year of the Linux desktop
Intuition

Linux keeps getting better, but the improvements never seem to have an effect on desktop market share. If we really want to reach unfamiliar users, maybe we need a different approach? BY JAMES MAWSON

Given the audience of this magazine, it hardly needs to be argued how much better, more beautiful, easier, and more useful desktop Linux has become in recent years. It's not like there's nothing left on the "to-do" list for a better and friendlier Linux, but there's also no denying how many items have been ticked off the list – and how many areas where Linux is already well ahead of the competition.

Even a few years ago, a Linux user needed to have enough technical skill to occasionally convert a file, download a codec, or update a driver. All these issues have almost totally gone away. The major Linux desktops are every bit as easy to use as macOS, and they are way more intuitive than Windows 10.

So has the Linux market share improved along with these significant improvements to the Linux user experience? It should vex us all that it hasn't. No matter how much we want Linux to break through to the mainstream market, it seems eternally stuck in the geek space.

Consider the case of the Valve Steam gaming platform. The Proton compatibility layer has brought thousands of new games to Linux [1], and the graphics drivers keep getting better, to a point where some native Windows titles now run faster on Linux. This night-and-day difference in Linux game support has lifted Linux's share of Steam users from under one percent to … under one percent.

So What's Missing?
Nonusers, by definition, are disconnected from the software. As I've become more engaged with this topic, I've become ever more fascinated to probe how far this disconnection goes. A big corporation that has a question about how their product is perceived by potential users typically commissions a marketing study, which could easily cost tens of thousands, or even hundreds of thousands, of dollars. But Linux isn't one big corporation.

I decided to do my own fact-finding mission. First I read through all the published data on what Mac and Windows users believe and know about desktop Linux. This was a very quick reading assignment; there is no published data.

So I spoke to as many nonusers as I could about Linux. I maintained this hobby as I traveled for work and pleasure, opening conversations with Irish backpackers, German school teachers, Albanian guesthouse owners, Israeli airport security, Thai scientists, and beyond. The most common take on Linux was "that's something to do with computers, right?"

Even many IT professionals do not think of Linux as a desktop OS: "Of course it's fast; the only thing on it that might slow it down is a text editor." Tragically, these types of outdated and simplistic responses even came from some admins who deal with Linux servers every work day.

What nonusers see of desktop Linux is contextual: social cues, media visibility, market position, perceptions, reputation, and so on. So no wonder the immense improvements in the software haven't driven adoption; to nonusers, these changes are invisible. The context they can see still says that Linux is a system for geeks.

Engaging Intuition
The Linux ecosystem has become quite good at speaking to the conscious, rational mind. If you want half-hour videos delivering depth and detail, they exist and are often excellent. But slow, careful thinking is cognitively expensive, so humans use it sparingly. We mostly default to much faster intuitions [2] that feel effortless and automatic.

Intuitive thinking makes rapid judgments from visible information and then immediately moves on. It's not tuned for objective truth and perfect choices but to navigate a busy and uncertain world with quick decisions that reliably avoid disaster.

When speaking to this fast style of thinking, the context is at least as much a part of the communication as what's said. So when the audience uses this surface-level stuff to decide that desktop Linux isn't for them, the code has no chance to change their mind.

Social Proof

By now, you can probably make a lot more sense of why Linux has done so much better across servers and embedded systems. It's not just that it meets these needs very well, though clearly that's a big part of it. It also engages quite excellently with the slow, careful thinking of the professionals paid to agonize over these decisions. To win with intuitive reasoning, we don't just need *more* marketing for Linux – we need a different kind of marketing. It needs to be alert to the biases and heuristics that characterize busy, distracted minds running on instinct.

The most important of these might be "social proof". This is a term coined by academic psychologist Robert Cialdini [3] to describe how people feel influenced to copy what they see others doing. We're more influenced when we feel uncertain, and we're more likely to be influenced by people we perceive as similar to ourselves [4].

It seems rational to suppose that, barring any technical impediments, the fact that professionals and power users choose Linux would drive its adoption with less technical users. Clearly, the pros know computers best, right?

A social proof perspective, on the other hand, argues that users will choose an operating system that they see others at a similar technical level using. If a consumer looks around and sees that users at similar technical level are only choosing Windows and macOS, it doesn't matter how many times you tell them about the stability and security benefits of Linux. As long as Linux is only visible as an OS used by the technically brilliant, the intuitive mind screams that it's too difficult for anyone else. If you understand this as a social proof problem, desktop Linux's adoption challenges look simpler yet more diabolical – it's unpopular for being unpopular.

Lenovo's Latest Love Letters to Linux

You might have caught Lenovo's recent run of announcements about extending Linux support to all their ThinkStations and ThinkPads [5]. It got quite a bit of coverage.

Of course this is a magnificent increase in choice for fully supported Linux PCs. It's also excellent social proof: the power of a global brand pushing a clear message through multiple channels. It is not just that many people see it; they intuitively realize many others see it too.

But where do we go with it? There is no reliable system in place for echoing this information out beyond the community of people who are already using Linux. And even Lenovo doesn't exactly help with this challenge. Their goal is selling hardware, not raising the general profile of Linux. Consequently, they say that the new support is targeted to "system administrators, AI professionals, IT engineers, and data scientists" [6] – highly technical roles – even though the potential market for ThinkPads extends far beyond this narrow range of users. Lenovo's announcement is still great news, but it falls well short of pushing desktop Linux into the mainstream. The announcement is thus a missed opportunity because there is no organization, movement, or even common understanding in place for how to project the information beyond the open source community and position the event in a way that would maximize social proof.

Where's the Leadership?

When Linus Torvalds said that fragmentation is "why the Linux desktop failed" [7], my hunch is that he was thinking like a lead developer, seeing code contributions diluted across so many projects. Fragmentation has also left nobody specifically in charge of communicating desktop Linux to a mainstream audience. There are segments of the ecosystem that engage in communication and marketing – some quite well. But these are efforts on behalf of individual projects or companies, and they are usually directed at existing desktop Linux users.

The "top of the funnel" marketing [8] – the initial awareness and interest – is a largely informal effort by independent media and word of mouth. There's a lot to like about this – there are talented communicators producing excellent content, but it is hard to see this effort producing a five- or six-figure budget for much needed things like market research and publicity. That's really more the work of an industry association.

Of course we already have the Linux Foundation, which would seem to be the organization best poised to push Linux out beyond the narrow sphere of technical users, but in recent years, the Linux Foundation has become even more focused on industry uses. A scan of the projects sponsored by the Linux Foundation shows a strong emphasis on cloud and server room technologies, along with AI, HPC, blockchain, and other industrial strength applications.

Even if you could convince them that driving desktop adoption is worthwhile, the sheer weight of competing priorities might make the effort a short-lived win. There's a glaring need for an industry body with a specific mission of bringing desktop Linux to mainstream audiences.

Commercial Forces

The conventional view is that industry as a whole is threatened by Linux and would somehow oppose more widespread adoption. That's probably quite true in places, but the IT sector has evolved in recent years, and several factors point to a need for change. Just because the major vendors don't want to lead the charge for desktop Linux doesn't mean they wouldn't benefit from it. Lenovo, for instance, like any Chinese PC manufacturer, surely worries about their reliance on a single American OS provider at a time of intensifying trade wars, so widening the Linux market is very much in their business interest.

Meanwhile, 95 percent of Steam users run an OS that increasingly pushes a competing digital distribution channel: the Windows Store. No wonder Valve has developed Proton. For these and other businesses facing disaster from the Windows monopoly, supporting desktop Linux is sensible insurance. Other Microsoft competitors aren't facing existential risk, but are well aware of the stark disadvantages of over-dependence on the Windows. Cloud services giants fall in this category.

Tier 1 hardware vendors of all nations might favor increasing Linux adoption simply because it puts pressure on the OEM price of Windows.

And of course, there are many Linux businesses that directly serve the desktop market too. For some combination of these businesses, a modest budget to promote desktop Linux is loose change. Someone just needs to organize it and put energy into explaining why it is worth the investment.

Toward a Desktop Linux Alliance

The point here is not to replace or sideline anything that anyone is doing already, but to take on crucial tasks that we're currently completely skipping. For instance:

Could We Have Some Market Research?

Our understanding of where mass audiences are is currently very vague and relies almost entirely on personal anecdotes. It's healthy to have opinions, but let's check them against the data.

Linux insiders who are brainstorming about how to market Linux often exchange ideas that sound plausible to each other. The problem is that even a robust debate can identify bad ideas. There are few protections against great ideas that make sense but still happen to be wrong.

For instance, there is a common complaint that desktop Linux hasn't gone mainstream because
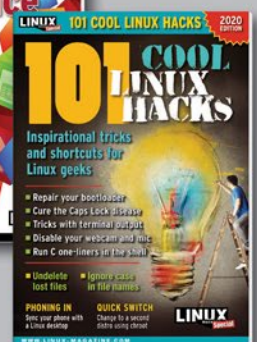
the options are too bewildering. Sounds plausible, but is it correct? How many Windows and Mac users could name even one Linux distribution or desktop environment? Who knows? We could keep arguing about it without it ever going anywhere, or we could gather the data.

Such data could, in the spirit of open source, be published for the benefit of everyone in our ecosystem.

Publicity and Influencer Engagement

Desktop Linux needs better social proof. A direct way to get it is to appear in mass media; not only will many people see it, they'll intuitively realize many others do too. That's where it would really help to have a great publicist on the case.

We also want desktop Linux to casually appear in conversation, as the iPhone does now. This could be nurtured with a content marketing campaign to promote Linux to bloggers, podcasters, and so on: not just in technology, but across all topics.

The Big Picture

The great tragedy of desktop Linux is that the software gets better all the time, but the social and psychological context has almost everyone deciding that it's not for them without even trying it. That's not meant to discourage or down-play the value of further improvements to the user experience and code. These things are always improving, because there's already a wide recognition that they matter, and Linux has great leadership to press ahead on the technical side. If Linux is to get a fair hearing with the world beyond its insular community, the social and psychological context – the only bit most nonusers even know – needs an equal measure of organization and leadership. If you want to read more of my thoughts on these topics, check out the Sunlight Manifesto [9]. There's also Make It Linux [10], a grass roots collective to promote desktop Linux to a wider audience in a way that engages intuitive reasoning. This is in a very early stage, and we're still figuring out what we're doing, but if you've got any ideas or want to get involved, please drop in to our Telegram group [11]. You can follow us on YouTube [12] and Twitter [13], too. ■■■

The Author

James Mawson is a technology copywriter, Linux and Raspberry Pi enthusiast, and lover of fried noodles. Visit his website at *https://www.handsomegenius.com.au.*

Info

[1] Valve's Proton project has brought 6,500 Windows games to Linux: *https://www.techspot.com/news/84909-valve-proton-has-brought-6500-windows-games-linux.html*

[2] Can We Trust Our Intuition?: *https://brainworldmagazine.com/can-we-trust-our-intuition-an-interview-with-dr-daniel-kahneman/*

[3] Robert Cialdini bio: *https://www.influenceatwork.com/robert-cialdini-phd/biography/*

[4] The Science and Secrets of Social Proof: *https://www.convertize.com/social-proof/#uncertainty*

[5] Lenovo Upping Their Linux Support: *https://www.linux-magazine.com/Online/News/Lenovo-Upping-Their-Linux-Support/(language)/eng-US*

[6] Lenovo press release: *https://news.lenovo.com/pressroom/press-releases/lenovo-brings-linux-certification-to-thinkpad-and-thinkstation-workstation-portfolio-easing-deployment-for-developers-data-scientists/*

[7] Fragmentation is why Linux failed on the Desktop: *https://www.tfir.io/fragmentation-is-why-linux-failed-on-desktop-linus-torvalds/*

[8] Top of Funnel Marketing: How to Get from Content to Leads: *https://blog.taboola.com/effectively-top-funnel-marketing/*

[9] Sunlight Manifesto: *https://sunlight.makeitlinux.org/*

[10] Make It Linux: *https://makeitlinux.org/*

[11] Make It Linux Telegram group: *https://t.me/makeitlinux*

[12] Make It Linux YouTube channel: *https://www.youtube.com/channel/UCOHfdVeTerUsUlbs4ezaEBQ*

[13] Make It Linux on Twitter: *https://twitter.com/makeitlinux*

Record and publish actions in the shell with asciinema
Shell Screencasts

Asciinema lets you record events at the command line and publish the resulting terminal movie on the web. BY CHRISTOPH LANGNER

A screencast (i.e., a movie of what is happening on screen) helps developers demonstrate their programs to users and is useful for people seeking a way of explaining their problems to a support specialist. On Linux, there are many different solutions for this, such as recordMyDesktop, OBS Studio, or – as in the case of Gnome – the feature is integrated into the desktop environment. But if you only want to record shell commands and their output, you're using a sledgehammer to crack a nut. Asciinema [1] can be a good, lean alternative for these cases.

Asciinema consists of three components. The first is the actual recording tool for the command line. The second is a web-based hosting platform for asciinema videos, which is similar to YouTube or image hosts like Imgur.com or Gfycat.com. The third component is a JavaScript player that plays the asciinema videos [2]. Users only need the recorder unless you want to host your asciinema videos on the web yourself; in this case, you would have to set up the server components on a web server.

Figure 1: Asciinema records the events on the desktop. The result is available as a special CAST file that can be played back via the terminal or a web-based player.

And … Action

Most current distributions include the screencast recorder for asciinema in their package sources. The application version counter is currently at 2.0.2. Ubuntu 18.04, Debian 10 "buster" (`sudo apt install asciinema`), and Fedora 28 (`sudo dnf install asciinema`) at least give you asciinema 2.0.0. More information about the installation, for example for the Python package manager Pip, can be found in the application documentation [3]. Having a recent version is important because the file format of the recordings changed with the release of asciinema 2.0 and many new functions have been implemented [4].

After the install, the easiest way to start recording is to type the `asciinema rec` command. Doing so opens a new shell in which asciinema records everything you type, and the system displays it on the screen. It continues until you stop recording by typing *exit* or pressing Ctrl+D. You can then either press Enter to upload the video to *asciinema.org*, or save it locally in the /tmp/ directory using Ctrl+C (Figure 1). The file name is always `tmp<random code>-ascii.cast`. Alternatively, you can pass a file name and path directly to asciinema at startup:

```
$ asciinema rec <example>.cast
```

To play the locally saved recording, call asciinema again, this time with the `play` option and the movie file as a parameter. A typical call looks something like this:

```
$ asciinema play /<path>/<to>/  ⤵
  <example>.cast
```

Asciinema plays the screencast directly in the shell, but without executing the recorded commands locally – it is in fact a video, not a script. If necessary, you can use the space bar to pause playback or press "." to skip through the video frame. Press Ctrl+C if you want to stop the playback of the terminal movie completely.

When you're recording, keep in mind that asciinema records the events on the command line unchanged. It not only stores the commands and their output, but if you correct input at the command line, this is also shown in the asciinema video. Similarly, if you do nothing during the recording, for example because you need to look up the details of a command, the video pauses.

To avoid what are often involuntary and (for the viewer) boring breaks, add the `--idle-time-limit=<seconds>` option or the shortcut `-i<seconds>` to the call to record the asciinema video. This limits the timeout to the number of seconds specified in the option. For example, `asciinema rec -i1` gives you a maximum timeout of one second.

Managing Shell Videos

Alternatively, you can upload the terminal video directly to *asciinema.org* on the web without any detours. You don't have to register with the service to do this; the video is uploaded automatically if you do not press Ctrl+C to cancel the action at the end of the recording.

Asciinema's terminal recorder then displays a URL in the style of `https://asciinema.org/a/j10[...]2wN`, where you (or others) can play back the movie in a web browser (Figure 2). The link remains active for seven days, after which the system automatically archives the recording, and access to the movie is lost.

If you want to keep the movie accessible for a longer period of time, you have to authenticate against *asciinema.org* on each computer where you want to create and upload recordings. To do this, enter the `asciinema auth` command and then open the link displayed in the terminal in a browser. You are then given the option to create an account on the asciinema website or to log in with an existing account. Asciinema will then automatically assign the terminal videos previously uploaded from this computer to your account.

The web portal (Figure 3) lets you manage your recordings. For example, you can change the metadata such as the title or insert a description. The terminal theme and the preview image can also be modified. *Settings | Make public* lets you publish your video on a list [5] accessible to all users. With the default setting, however, your uploads remain private. Unsuccessful movies, or ones you no longer want, can be deleted from the overview.

Embedding asciinema

The *Share* button gives you information on how to integrate the selected terminal video into any website, such as your own WordPress blog or the one of your projects' GitHub page. The page auto-
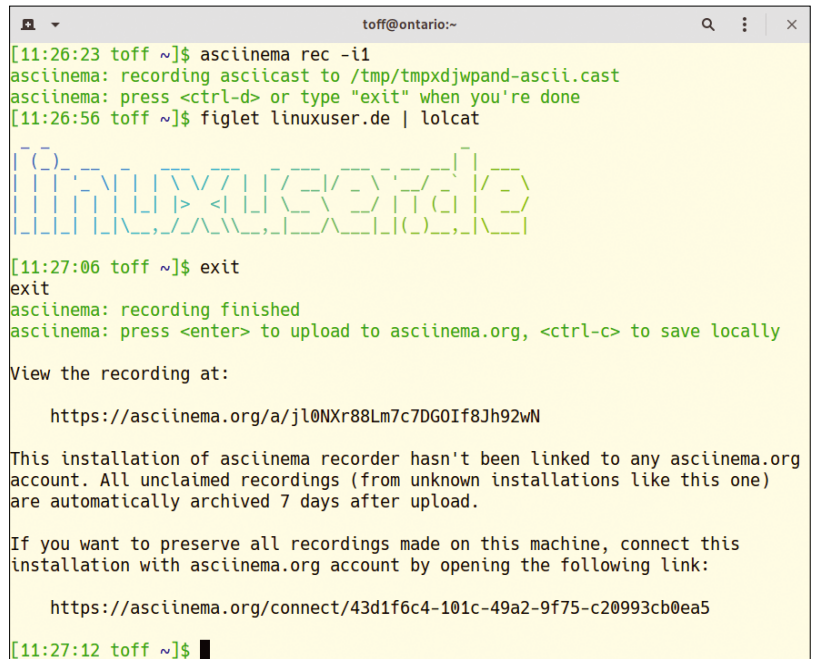

Figure 2: In the default configuration, asciinema automatically uploads the recorded movies to the in-house "YouTube alternative" for terminal movies.

matically generates HTML or Markdown tags that display an image in the web page linked to the video on *asciinema.org*. You will also find a short script snippet below, which can be used to embed the video directly into a website (if the content management of the site allows it).

Listing 1 demonstrates an asciinema-enriched website. The first section uses the script; the second uses the universal image. Figure 4 shows the resulting and still very rudimentary website in the browser. The asciinema movie played back in the web browser supports copy and paste actions at any point, so that the viewers can, for example, copy commands from the video directly into the terminal.

Figure 3: Asciinema's web portal lets you organize the terminal videos associated with your account. You can adjust the appearance of the thumbnails and the web player.
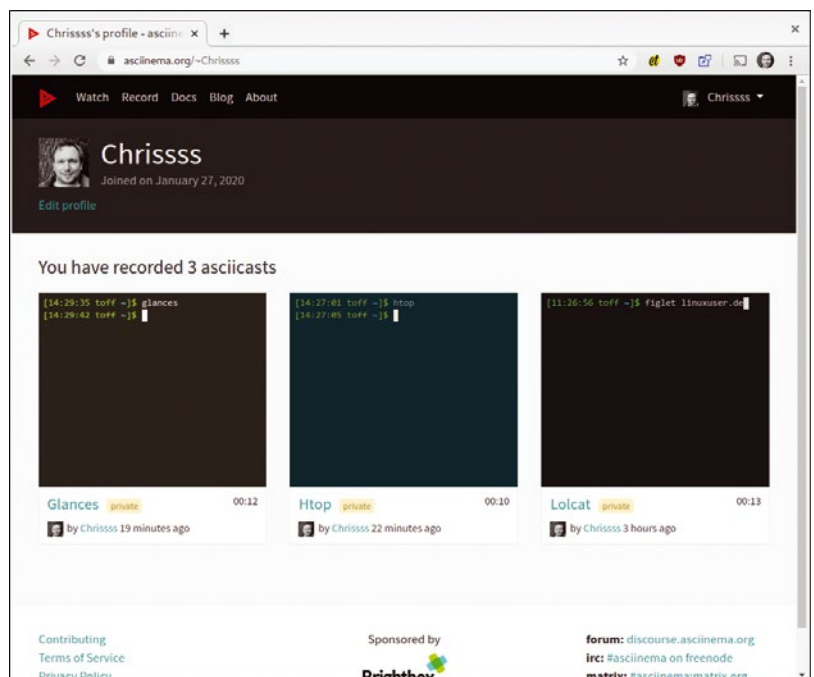
Listing 1: asciinema-Enriched Website

```
<!DOCTYPE html>
<html>
  <head>
    <title>Hello World</title>
  </head>
  <body>
    <h1>Included as script</h1>
    <script id="asciicast-jl0NXr88Lm7c7DGOIf8Jh92wN" src="https://asciinema.org/a/
            jl0NXr88Lm7c7DGOIf8Jh92wN.js" async></script>
    <h1>Included as image</h1>
    <a href="https://asciinema.org/a/jl0NXr88Lm7c7DGOIf8Jh92wN" target="_blank">
            <img src="https://asciinema.org/a/jl0NXr88Lm7c7DGOIf8Jh92wN.svg" /></a>
  </body>
</html>
```

However, both variants have the disadvantage of embedding external resources into your own website (the script and the data loaded by asciinema or the embedded image). To avoid external data connections, you will therefore want to download the image of the terminal movie, upload it locally to your website, and then link only to the asciinema video. It's also possible to convert asciinema clips to animated GIF images using other applications (see the "Alternatives" box).

Conclusions

Asciinema is primarily recommended for developers who want to demonstrate scripts or terminal-based applications. If you use the services of *asciinema.org*, you can publish the movie faster than you can record it. But other user groups will also benefit from easy-to-create movies. It helps you to communicate what is happening on the command line quickly to other members of a Linux forum. ∎∎∎

Figure 4: The terminal videos hosted by asciinema can be easily integrated into your own web pages, such as a blog or GitHub page.

Info

[1] asciinema: *https://asciinema.org*

[2] asciinema project on GitHub: *https://github.com/asciinema*

[3] Installation instructions: *https://asciinema.org/docs/installation*

[4] asciinema blog: *https://blog.asciinema.org/post/two-point-o/*

[5] Public list of all asciicasts: *https://asciinema.org/explore/public*

[6] asciicast2gif: *https://github.com/asciinema/asciicast2gif*

[7] gifcast: *https://dstein64.github.io/gifcast*

Alternatives

As alternatives to asciinema, asciicast2gif [6] and gifcast [7] let you convert asciinema clips to animated GIF images. In this form, the video can be integrated on websites that will not let you embed third-party scripts and thus the asciinema video. While you have to install asciicast2gif locally, gifcast is also available as a web application that converts your videos without needing to install any programs on the local computer.
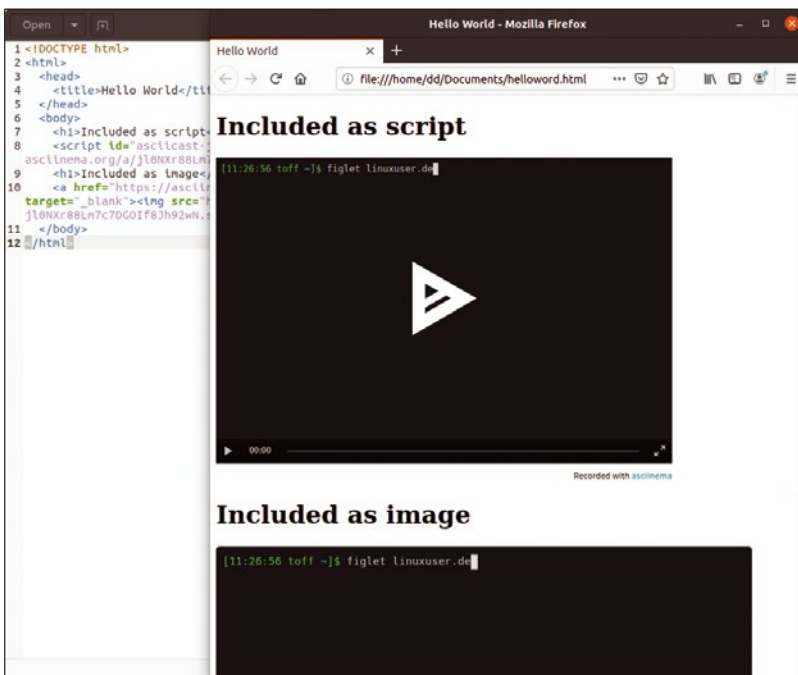
FOSSPicks

Sparkling gems and new releases from the world of Free and Open Source Software

Graham has been brewing his own beer this month thanks to an old Raspberry Pi and an even older refrigerator for temperature control. To make it more exciting, it's brewing under 15 PSI of CO_2 pressure! BY GRAHAM MORRISON

Audio workstation

Ardour 6.0

Ardour is a hugely important piece of open source software. It does for audio what Blender does for modeling and 3D animation, what Gimp and Krita do for image editing, and what Kdenlive does for video editing. Without it, we wouldn't have a single professional-grade audio recording and mastering tool, and we wouldn't be able to complete any of those 3D animation/film/movie/music production chains on Linux. To finish any media project, you'd need to use a different operating system. And so would the thousands of users and institutions who rely on Ardour to teach professional audio production while saving the considerable outlay needed to buy a commercial equivalent. Ardour makes audio production accessible.

Version 6.0 is a major update to Ardour and comes four years after the previous major release, Ardour 5.0. Its GTK+2 UI remains in place, because all the internal audio processing has been overhauled, which is far more important than the already streamlined aesthetics. This audio overhaul has been a huge and complex undertaking by Ardour's small development team, and it has touched almost every internal aspect of how audio is handled, from the algorithm used to mix everything together to the sample accurate way latency variation has been removed from all buses, tracks, plugins, sends, inserts, and returns. All of which makes Ardour sound considerably cleaner and more accurate, more predictable, and more like other software like Pro Tools, especially on large projects with dozens of tracks, plugins, and buses.

Another major feature that improves the audio workflow is global varispeed. This allows you to tweak the playback speed of all the audio you record. This is commonly used as you would a tape machine to hit specific target points in video or to readjust for timing differences among recordings. The resampling algorithm used to accomplish this magic sounds fantastic. You can also finally monitor both an audio input and the recorded audio on the destination track at the same time, which is vital for overdubbing or for recording a different take of something.

There's also a huge number of smaller features that will have a big impact on your music making. Ardour finally supports PulseAudio natively, for instance, and you can concurrently use different ALSA input and output devices, which means you can finally avoid using JACK to keep things simple. You can even record directly to FLAC to save space. There's an onscreen MIDI keyboard, complete with keyboard shortcuts, and the MIDI note editor now shows the note's velocity with a gradient. Plugin support is still excellent, and you can now create tags to better manage your collection. The inline graphics for Ardour's own effects have improved, making this a brilliantly unrivaled feature for quickly glancing over your mix to see compression levels and EQ targets. As too is the plugin DSP load viewer, which now shows which individual plugins are taking up your system resources, alongside their specific minimum, maximum, average, and standard deviation response times! Like Blender, Ardour has a steep learning curve, but the end result isn't an open source compromise in any way. Ardour is capable of taking serious, professional recordings through mastering to production – all within its uniquely flexible routing system.

1. **Plugin tags:** Tag your many plugins for quicker access. 2. **Monitoring:** Finally, you can now monitor both the recording and the track input at the same time. 3. **Inline plugins:** See compression, expansion, spectrograms, oscilloscopes, and filter curves directly in the channel strip. 4. **MIDI keyboard:** Click or use keyboard shortcuts to generate MIDI data. 5. **DSP load:** See exactly which plugins are adding latency and which are using the most system resources. 6. **Varispeed:** Change the playback speed and everything is now automatically resampled. 7. **Wet recording:** Move the Fader item in the channel strip to record the wet (effects) up to that point in the signal chain.

Project Website
https://ardour.org

3D astronomy
Celestia

Almost 20 years ago, an astronomy application appeared that completely broke the mold for all other astronomy applications. At least on home computers, there had only been enough processing power to calculate a static view of the night sky from a single position at a single point in time. If you had the resources, you could update this view to track time. But Celestia unshackled observers from their backyards, and even from Earth's gravity, by allowing them to fly through space and time to observe a first-person view from anywhere in the universe. All of this was done in real time via the wonders of 3D acceleration and OpenGL, and it allowed you to see the universe from a whole new perspective – from Comet Shoemaker-Levy 9 hitting Jupiter in July 1994 to the solar eclipse that will happen over Easter Island on Monday, December 14, 2020.

Celestia has been in stasis since 2011. During the interim, other applications have come along offering similar functionality. Perhaps the best is the proprietary Universe Sandbox, which allows you to play with celestial objects and the laws that keep them in motion. Even games like Elite Dangerous have borrowed heavily from Celestia's exploratory premise. Which is why it's so brilliant to see that the Celestia project is showing sparks of life again, with new builds and a new nightly release. The forums are alive, too. While the current Celestia experience is very much the
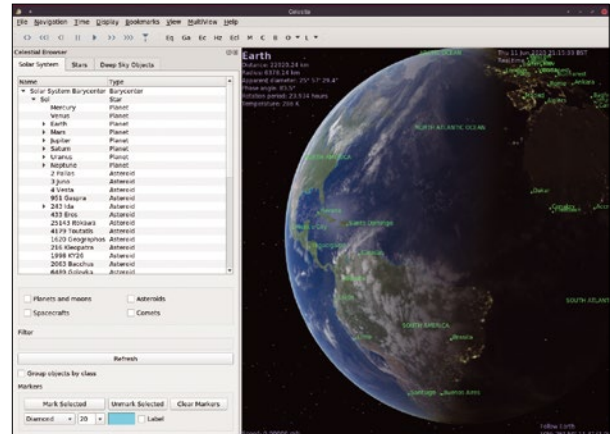
Even after so many years, Celestia still offers some unique perspectives on the universe, and it's slowly coming back to life.

old Celestia experience, it's wonderful to see it running in the modern age, and it hasn't aged badly at all! This is helped massively by the huge texture and add-on packs you can install. These upgrade the relatively simple geometry with textures of which more recent hardware will make good use. There are still plenty of features unique to Celestia: split views, bookmarks, easy time shifts, and even support for 120 frames per second!

Project Website
https://celestia.space/

Liquid cooler control
liquidctl

OpenCorsairLink is one of those small yet infinitely useful utilities we cover in these pages. It allows us lowly Linux users to communicate with our Corsair power supplies and coolers and to perform essential tasks like lowering their fan speeds, adjusting the pump rates, and changing LED colors to pink. Without any support from Corsair, the developer behind OpenCorsairLink used Wireshark to completely reverse engineer the USB communication between the official Windows client (Corsair Link) and the real hardware. It became too much for a single developer, who has announced the retirement of the OpenCorsairLink project, but the good news is that its original author has already migrated many of its core protocols to another project, liquidctl, a similar initiative but with a larger team and a more open premise.

Unlike OpenCorsairLink, which is written in C and bundles everything together, liquidctl is written in Python and abstracts the drivers that communicate with the hardware from the functions needed to configure it. This means you can use the same syntax to set the fan speed profile for a ZXT Kraken X53 as you would on the Corsair H80i GT. It also means that when other hardware gets driver support, it will immediately be able to take advantage of the functions provided by liquidctl. This already covers most of the functionality found in OpenCorsairLink, and even lets you change the colors of the LEDs. You can also get verbose status output from your devices, which is
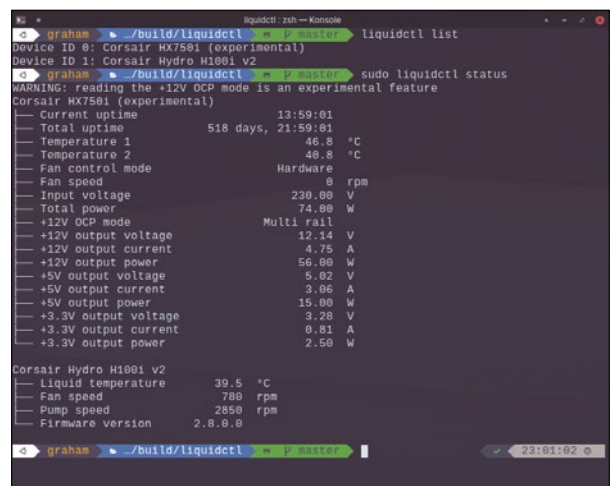
OpenCorsairLink is dead. Long live liquidctl, which will hopefully offer similar functionality for many more devices.

often the only way to see temperatures, fan speeds, rail voltages, and pump speeds from Linux. Another great feature is that because each driver has to be implemented separately, each device has its own documentation to deal with its capabilities and quirks. This makes it easier to set up than OpenCorsairLink and easy to integrate into your system as part of systemd or your startup routine.

Project Website
https://github.com/jonasmalacofilho/liquidctl

Telegram client fork

Kotatogram

The Telegram messaging platform is one of those projects that has benefited hugely from recent global upheaval, mostly thanks to simply not being WhatsApp. This is because it implements group chat, instant messaging, telephony, end-to-end encryption (not by default), and a smorgasbord of animated GIFs and memes in much the same way as WhatsApp, while being somewhat open source and not owned by Facebook. Signal is a better open source/security/privacy option, but Signal doesn't yet offer the same level of WhatsApp style integration for those trying to convince family and friends to try something other than a platform owned by Facebook. While Telegram's server-side protocol isn't open source, the client and its protocols are, which leads to all kinds of different interpretations. This is another good thing, because the official Telegram Desktop client also has some configuration shortcomings, and that's exactly what this fork attempts to address.

As a fork, Kotatogram behaves exactly like its upstream project. This means it authenticates in the same way and still requires a telephone number for new members. At the moment, it also behaves exactly like the original project, but there are signs of development with its own set of configuration options. You can finally use a custom font, for example, or compact chat lists. Sticker sizes can finally be changed, as can the chat bubbles. These are the kind
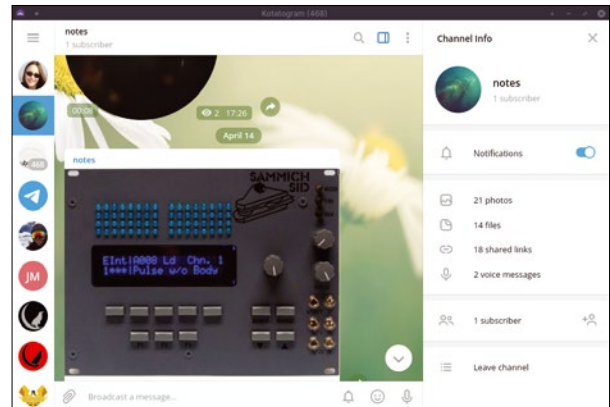
If you're looking for a version of Telegram Desktop with more configuration options, take a look at this fork.

of updates that will never make it into the official client, and the Kotatogram maintainers have been quick to respond to feature requests and keep an updated target list of many other potential quick fixes. The end result is an app that's just as easy to use as the original, with code you can check yourself for backdoors and malware, but that may allow for more dynamic options than the upstream version. All we need to make it perfect is an open version of the server software.

**Project Website**
https://github.com/kotatogram/

Audio ripper

Vitomu

This fantastic little application for pulling audio from online video could be used to make an illicit copy of copyrighted material, which we genuinely don't want to encourage or obviously endorse. But it also has legitimate uses, especially when online content has become so ephemeral, often disappearing without warning. It's also a brilliant example of great design, because it performs a function that is either complicated to do manually, unreliable, or advertising-ridden when you use an online service. Vitomu simply waits for a video URL to appear on the clipboard; when it does, it lets you click a single button to convert the video at the end of the URL into an audio file. It's perfect for straight-to-video podcasts, and it works without requiring any complicated dependencies. Vitomu will even download the latest version of FFmpeg internally when you first start up the application.

While Vitomu performs its function without requiring a single click, there are some configuration options to explore. There's a range of audio formats to choose between, including FLAC, Ogg Vorbis, M4A and AAC alongside the default MP3. You also can change the bitrate. The minimalist look can be tweaked slightly, first by changing the font size and also by enabling a custom titlebar, though the effect is a subtle movement of the text if you're using a dark theme. The highlight color can be switched between pink and green, but either way the application always looks fantastic. It may seem
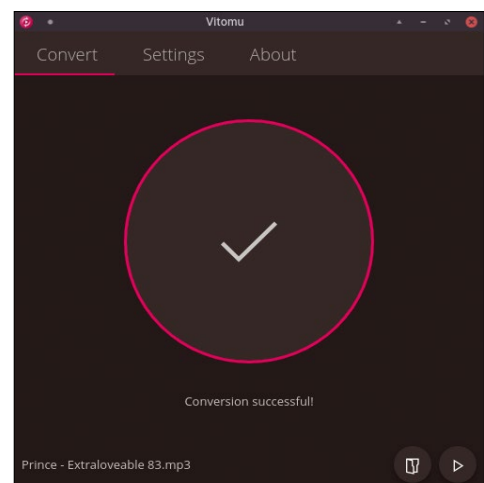
There are websites that rip audio from online video, but it's far better if you can keep that sort of process local with a tool like Vitomu.

like a simple task, but having one application that performs the job well, and transparently, without being another cookie gathering web page, can be a huge advantage, because you'll never need to remember the equivalent FFmpeg incantation or web address again.

Project Website
https://github.com/digimezzo/vitomu

Advanced note-taking
Knowte

Knowte is a note-taking application by Raphaël Godart, the same developer who created the Vitomu Internet video-to-audio converter also covered in this FOSSPicks. This is a good thing, because Vitomu is beautifully designed, and fortunately, Knowte borrows the same aesthetic. It's also a much more ambitious application that offers more than you'd typically expect from a note-taking application.

The first thing you notice is that there are five headings across the top of the main window: *All*, *Today*, *Yesterday*, *This week*, and *Marked*. Each heading embeds a counter that shows you how many notes you've taken for each period. To create a note, you first need to create a notebook. This is like

a folder in filesystem terminology, and it allows you to create notes for different categories, although you can also select *All notes* to show every note regardless of which notebook they're in. All these notes are also part of a collection, and you can create a new collection for an entirely new set of notes. This could work well for class notes, for example, or for taking notes during a project that are then archived as an old collection.

The notes themselves are written from a separate window that operates like a full-fledged text editor. There's a toolbar with formatting options, and you can use Markdown square bracket syntax to create checkbox items. The number of checked and unchecked items in this list is visible
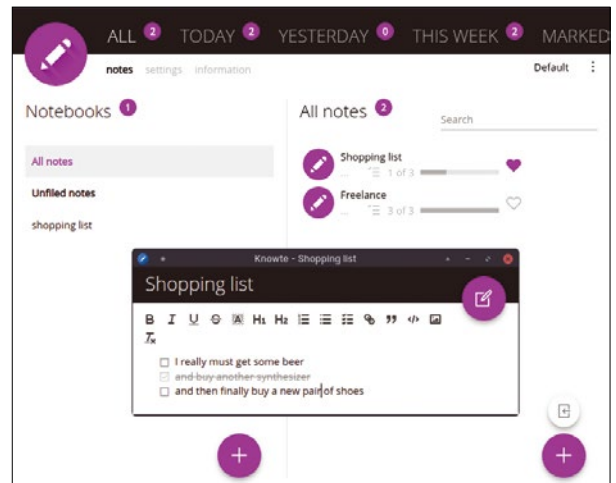
Knowte is more like a note-taking IDE than a simple editor app, letting you create and manage your personal notes with ease.

outside the note within the *All notes* view, so you can easily see the progress (or lack thereof!) of any to-do lists you've set yourself. It works brilliantly, and the design is always clear, concise and easy to understand. Notes appear to be stored as plain text within a JSON-formatted file tracked by a simple database. It's not as portable as simple Markdown, but it still means you could easily get to your data if necessary, or if Knowte ceases development.

Project Website
https://github.com/digimezzo/knowte

Simple note taking
Notorious

As demonstrated above, there are many note-taking applications – from cloud-synergized to-do lists through to Vim add-ons. This is for two reasons: They're easy to implement, and they're eminently useful. This usefulness is also directly related to how easy you find them to use, and how much they make you want to use them. Two of our favorites are console-bound Vim-Wiki and TodoList on the Plasma desktop. Notorious, though, is another beautiful and minimalist note-taking app that also has a relatively unique feature – simple input with a keyboard shortcut for almost everything while remaining low on resource use and perfectly integrated with any Gnome desktop.

A list is created by entering a name in the search bar and pressing Return. You can also navigate between your saved notes by using the cursor up and down keys, and the Escape key switches between edit mode and note select mode much like command mode in Vim. This allows you to very quickly create a note when the application has launched or find any note you've already made. You're free to format your lists as you please, although if you stick to Markdown syntax, its keywords will be highlighted after enabling a configuration option. This is useful for highlighting subheadings, italics, and bold sections of your lists. You can also insert emoticons, and your notes are automatically saved when you quit the app.
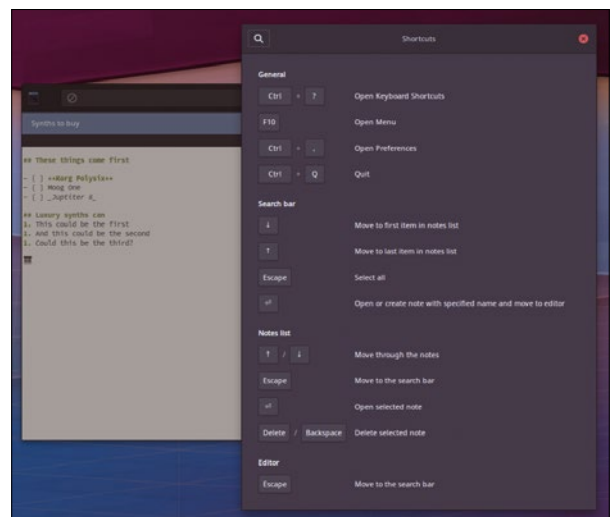
The best note-taking apps get out of your way and help you quickly jot down your thoughts.

With only a handful of functions, it's easy to remember the keyboard shortcuts, which are listed by pressing CTRL+?. There are very few configuration options, but you can switch between a light and dark theme, and you can also choose where to save your notes by default, as you're not asked to manually save at any point.

Project Website
https://gitlab.gnome.org/GabMus/notorious

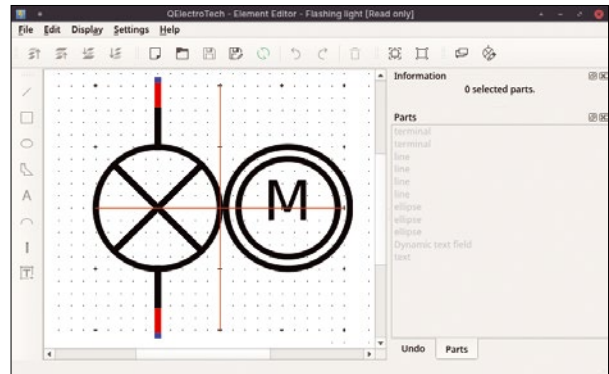Circuit illustrator

QElectroTech

O ne of our most interest-ing recent finds was Ho-rizon EDA, a circuit and printed circuit board (PCB) de-signer with a huge component database that you could use in your own designs. It even fea-tured a fully split 3D breakdown view of your designs and could produce final output ready for production. But these advanced features make Horizon a bit in-timidating, especially when your needs are a little less ambitious. In which case, QElectroTech might fit the bill. Unlike the full-fledged feature festival of Eagle, QElectroTech is more of a plan-ning, design, and illustration tool tailored to suit circuits of all kinds. This is fantastic if you need help adding diagrams to a paper or article. It's even better if you want those diagrams to be functional, because it can also help turn those designs into a circuit for your final project. What's more impressive is that it isn't specifically targeted at small-scale electrical circuits, but at a great swath of electrical and even mechanical projects, all from a single application.

To accomplish this flexibility, QElectroTech bundles its own large collection of what it calls elements. These elements are the graphical components you drop into your blank canvas de-sign; they're listed in the right-hand pane of the main window. Categories for these elements include logic, hydraulic, pneu-matic, energy, water, refrigera-tion, and solar, and there are a huge number of elements to use. Placing them into a new project works in much the same way as using a design application like Inkscape. Se-lect your tool or component, drag it onto the canvas, and line it up with your other ele-ments. Electrical terminals are clearly marked in red, and the grid snapping helps you keep everything on the same axis. When using the drawing func-tions, such as the line tool, the

> ...it's one of the best ways we've seen to illustrate and document plans from circuits, as well as the various mechanical projects at all kinds of scales.
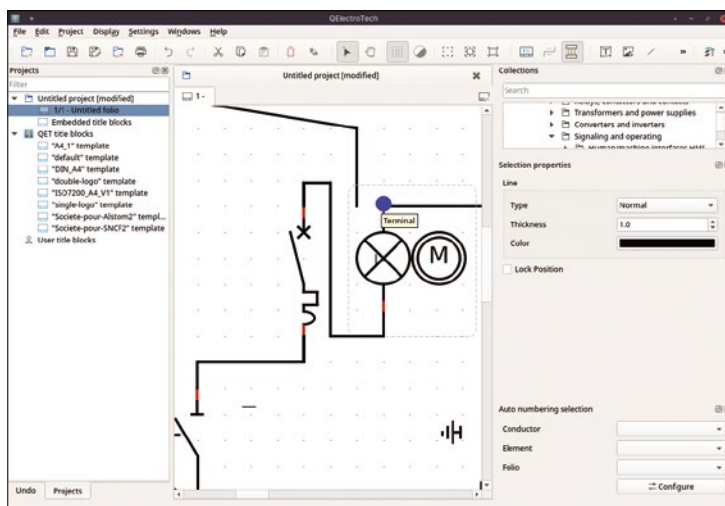
terminals will automatically be-come highlighted and easy to select so you can create cir-cuits accurately. You can also

connect terminals together by dragging from one to another. The circuit will automatically adapt to your layout to fit the connection into your design, which is a great feature for cre-ating flow charts.

The background canvas is called the folio; opening its prop-erties page allows you to change how elements can be arranged. It's here, for instance, that you can set the number of columns and rows for the grid, as well as their size. You can also set the angle for the text you add, as well as its font and size. All of this can be saved as a template, along with an automatically generated folio title block that can include a title, author, date, location, and page number. Another of QElectro-Tech's best features is the ele-ment editor. This allows you to create your own components using a new set of drawing tools, including polygons, circles, lines, rectangles, and text. You can also add the terminals. These are used to position the component in the main window, and to automati-cally connect your component with others in the circuit. It might not include any of the IDE-like tools you might need when test-ing or simulating a circuit design, but it's one of the best ways we've seen to illustrate and document plans from circuits, as well as the various mechanical projects at all kinds of scales.

When finished, the design can be exported as an image or SVG or DXF model. You can also generate a separate parts list.

There's a lack of general diagramming and flowchart tools on Linux. QElectroTech is more than capable of performing either of these functions.

Project Website
http://qelectrotech.org/

Terminal Scorched Earth

Gorched

Gorched is an interpretation of Scorched Earth, an old PC game dating from 1991. Even if you've never heard of it, it's likely you've already played one of its many clones. It's a game style most famously known simply as "tanks," or "Atomic Tanks," "Xscorch," or perhaps even as the world-conquering "Worms," although the latter is the ultimate extrapolation of the original idea. The gameplay is simple. A 2D landscape is first randomly generated, featuring large hills and low valleys, and two opposing tanks are dropped onto the land, one on the left and the other on the right. The player then typically presses space to initiate a shot. The angle of the tank's gun, and the eventual velocity of the projectile, counts up at increasing speed. You need to

get the timing right and press space again to fire. You then watch as your projectile flies through the sky, hopefully hitting the rival tank close enough to make it an easy adjustment for the next shot. However, before you can try again, you need to survive a shot back from your opponent whose turn it now becomes to blast your tank off the face of the landscape.

The brilliant thing about Gorched is that it accomplishes all of this from the humble command line using almost pure ASCII text characters. The colors are bold and easy to see, and while the tanks only show the changing angles through text, launching the projectile is as fun and as addictive as in the original. What's even better is that the landscape is destructive! This means when you
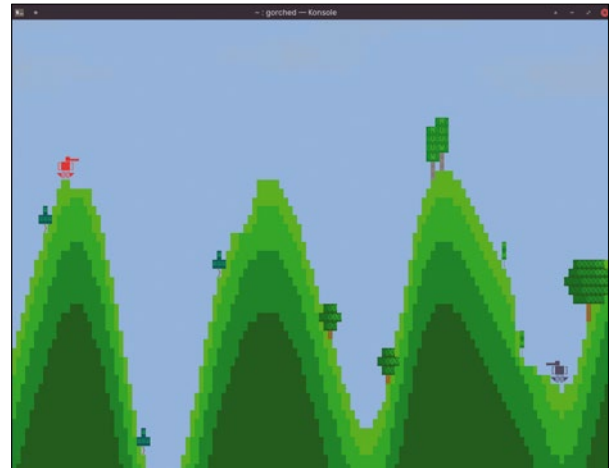
Play a two-player Scorched Earth clone in ASCII on your terminal, probably even remotely using tmux, although we didn't try this!

miss, you destroy part of the background landscape, which can even collapse along with the little ASCII trees. There's currently no AI, so you need to play both sides if you're on your own, but this means you can get some sneaky practice for when you do convince someone to play.

Project Website
https://github.com/zladovan/gorched

Platformer

PACE

PACE is a game at an early stage of development, with its initial release coming after a mere five days of development and with only 10 levels. But it's quickly becoming a mature and fun little strategy game, which now includes 35 levels, its own sandbox mode, and even a tutorial mode for beginners. The game itself features super-simple monochrome 2D graphics, with a stick person as your player avatar. Each level is a maze or cave viewed from the side, and you use WASD or the cursor keys to move your avatar around, with the Shift key used to add a little extra speed. This speed-up is limited by a percentage countdown that will only give you a boost for a short period of

time before recharging. As the game's name suggests, it's this speed that is the important bit. To get to the next level, you need to get your player through the maze to the door within a certain amount of time.

Levels usually consist of multiple screens, meaning that you need to quickly get across a couple of screens to trigger a colored switch that opens a door before heading back to escape. Multiple switches, rotating walls, and floating platforms all add to the complexity, and you often need exacting platformer skills to negotiate the gaps and platforms at top speed before a door closes. The early levels are moderately challenging, but they get difficult in later stages. Much like other
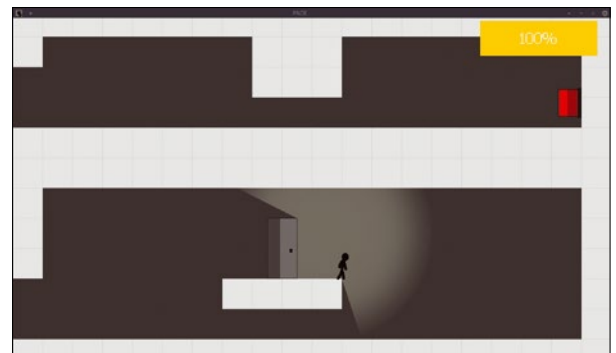
PACE is a super-quick platformer where you run against the clock to open and exit through a single door.

super-tough retro platformers, though, it's addictive because you can restart a level so quickly and replay through to where you keep getting stuck without everything becoming a little too arduous. And if it ever does become too arduous, using the built-in sandbox/level editor to create your own challenges would be a great way to contribute back to the PACE community.

Project Website
https://simonboi.tech/projects.html

Tools for collaborative office work

Working Together

Small businesses and organizations can benefit from products that expand on basic office suites to support collaborative work and that can be integrated with other services to suit individual business needs.

BY ERIK BÄRWALDT

While complex systems such as customer relationship management and enterprise resource planning software are often used by larger companies and organizations, advanced office packages with collaborative functions are more suitable for small organizations.

This developing market supports a growing ecosystem of applications that are based on basic office suites and that enable extensive teamwork. Depending on the needs of your office, the functionality of these packages can also be extended with options that include integration with other applications and services.

The focus here is on ease of use. A further focus of development under Linux is on increased compatibility with a wide range of file formats. In order to comply with established standards, the goal is the best possible conversion of the widest variety of formats.

This article looks at products from two organizations, Collabora Productivity and OnlyOffice. Both companies offer cloud-based office suites, as well as infrastructures that rely on local desktop installation and on-premises servers (see box "IT Infrastructure").

Collabora Products

Collabora Productivity [1], based in Great Britain, focuses on the enterprise market. Collabora Office is an office suite available for local desktop installation, and a package named Collabora Online is for cloud-based use.

However, both packages offer some additional flexibility. Collabora Online can also be used in an on-premises environment. It supports Nextcloud, ownCloud, Seafile, Pydio, and FileCloud environments, and the Univention Corporate Server also works with Collabora Online. For Collabora Office, two versions for mobile operating systems are available so that smartphones and tablet PCs can be integrated into a collaborative network.

The Collabora Online Development Edition (CODE) is a third option based on the Collabora Office core. It is offered as a developer version for on-premises solutions with Nextcloud or ownCloud.

IT Infrastructure

In many companies, collaborative work in heterogeneous IT infrastructures has been the norm for years. However, different user needs and varying workplace requirements have created a very confusing market. In principle, three forms of IT infrastructure are suitable for modern cooperative workflows: cloud-based services, dedicated on-premises client-server infrastructure, or web-based applications that also access on-premises servers.

While the web-based services run in the browser and therefore do not require specific applications client-side, on-premise solutions, with their own clients and servers, are usually more complex to implement because they often include applications on the client computers. In contrast, cloud services are offered by various external service providers, who also manage the complete infrastructure as a Software-as-a-Service (SaaS) solution on a subscription basis.

On-premise solutions, on the other hand, benefit from the fact that they do not outsource the data and can therefore also implement user-specific data protection guidelines more easily. In addition, depending on the scope of the collaborative applications, they often prove to be more cost-effective than a cloud-based service in the longer term. Cloud services, on the other hand, require hardly any administrative support.

The online versions are intended for web-based client use – on the client computer, you only need a current web browser to edit and manage documents of any kind.

The underpinnings for all Collabora products are LibreOffice, which Collabora expanded with additional functions for collaborative work in a cloud environment. The company also aims to improve on LibreOffice's conversion filters for various formats.

The development edition is available for free, and the company offers the other Collabora products as part of a subscription in two price models based on the number of users [2]. Collabora assures long-term support for all commercial products. Discounts are available for multiple-year subscriptions, and there are options for support services and integrations that can be provided by local partners. Trial versions of the different cloud-based packages are available on the Collabora website.

Collabora Online

Collabora Online (Figure 1) integrates the modules Writer, Calc, and Impress from the LibreOffice package, so it includes the three most important office applications: word processing, spreadsheets, and presentations. The less frequently used modules Base (database) and Math (formula editor) are not included. For cross-platform use, the suite not only supports the Open Document Format (ODF) but also all newer Microsoft formats, which makes interaction with other office suites easier.

The editing functions of the individual modules are sufficient for the majority of daily office tasks. For example, scientific texts can be edited with integrated footnote and endnote management, and the suite also includes many templates.

Unlike the locally installed version or LibreOffice, there is no form management in Collabora Online, and this version also restricts the table functions available in word processing. However, it does include graphic elements, such as a numbering function and drawing lines, and users are able to add short text comments.

Compared to the locally installed office suite, the response speed may vary significantly when using the external cloud solution due to bandwidth fluctuations.

Importing and exporting third-party formats is not always a smooth experience with Collabora Online. Although the suite usually loads documents without any problems, in many cases some work is required to get imported files back into shape.

Layout errors occur even in ODF documents if the same fonts are not available on both systems, so using standard fonts is recommended. Collabora also does not execute Microsoft macros when importing online documents.

You will also want to use a different software solution for viewing PDF documents. As with other office suites, Collabora's import filters frequently cannot handle complex PDF documents.

Collabora Office

The desktop package is called Collabora Office, and it is available for Linux as a 64-bit version for

Figure 1: Collabora Online, based on LibreOffice, includes word processing, spreadsheet, and presentation modules.
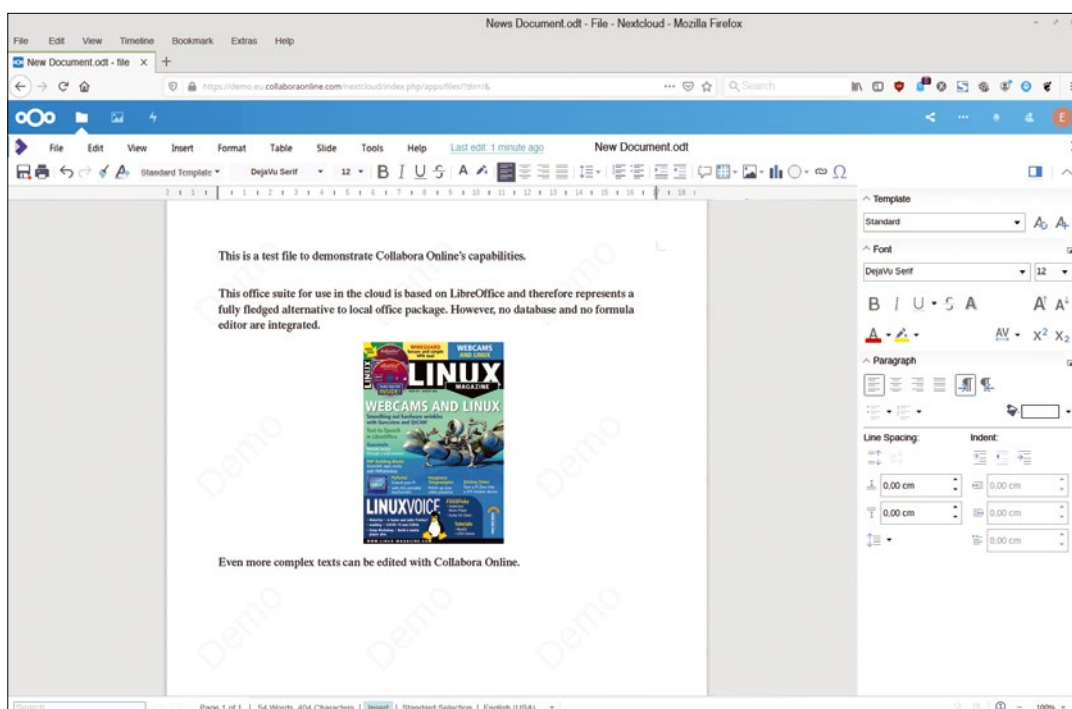
distributions with deb and RPM package management. Localized help and language files are available for different countries.

Unlike the online version, the office suite for local installation also includes the Base database module and the Math formula editor. The office package also supports other third-party formats for which Collabora Online does not provide conversion filters. Visually and functionally, there is practically no difference between Collabora Office and LibreOffice.

Whether office solutions work in heterogeneous IT environments depends largely on the file exchange options, and the Collabora website highlights the quality of its conversion filters as compared to LibreOffice. However, when testing, we did not see improvements in document layout and formatting when importing the same files into Collabora Office. For example, the filter loaded slides saved in ODP format incorrectly for the most part, and even more sophisticated texts in Microsoft formats had a partially incorrect layout after being imported.

As with LibreOffice, the collaborative functions in Collabora Office include functions for storing and loading files on remote computers. You can add comments to any document for later editing. The comments appear as colored notes in a column to the right of the page view, with a line pointing to the associated text, and you can identify comments from individual users.

EGroupware and Collabora

Another option you have with Collabora is to use it with EGroupware (Figure 2), a product from the company of the same name, which is based in Germany. The basic version of the EGroupware package [3] is a comprehensive solution for collaborative work that seamlessly integrates Collabora Online for a fee [4]. There are several commercial editions that include cloud access with different storage capacities. Alternatively, you can also use the EGroupware suite with on-premises servers in a local cloud.

EGroupware provides integrated solutions for calendar, address management, email, task management, project management, and messenger functions. The suite also contains a file manager, rights management, and various synchronization, import, and export filters.

In some cases, EGroupware and Collabora Online complement each other in the accomplishment of tasks. For example, data from EGroupware can be used to fill out documents in Collabora Online without separate manual preparation. Both packages are interlinked via the EGroupware admin panel.
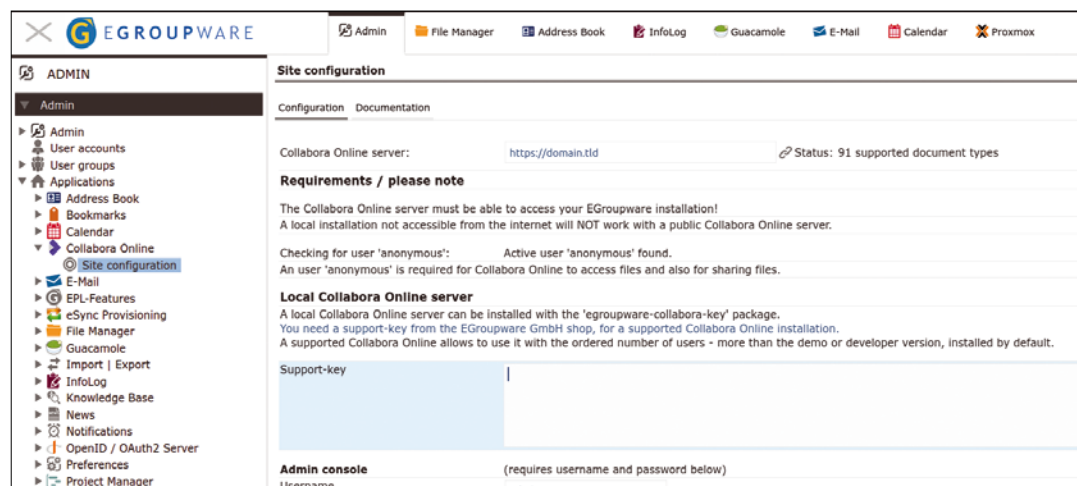
OnlyOffice

OnlyOffice [5] was developed by Ascensio System SIA, based in Latvia. Like Collabora, it is available in both cloud-based and locally installed versions.

The focus is on online editors. The Integration Edition is suitable for ownCloud and Nextcloud, but also for platforms such as Alfresco or Confluence. The Enterprise Edition, which is available in several versions, includes the entire range of functions of online editors, as well as a wide variety of tools for everyday office work.

You can host these packages on your own servers. An additional Developer Edition lets you integrate the core components of OnlyOffice into your own applications. These commercial versions are available under proprietary licenses.

The company also offers a free Community Edition of the Enterprise Package, which includes all the basic services but limits access to a maximum of 20 simultaneous connections. The Community Edition is also used in the context of an on-premises installation. It is licensed under the free AGPL license and has a limited support offering.

Figure 2: EGroupware offers great integration of Collabora Online.

For Linux, a Docker image and several other options are available for download. In addition, there is a locally installable version of the suite with OnlyOffice Desktop Editors and two additional packages for smartphone operating systems. Even the locally installable office packages are not derived from a free office package, but are a proprietary development with its own user interface.

Online Editors

The central component of the online OnlyOffice solutions are the online editors [6]. They include the same word processor, spreadsheet and presentation modules, regardless of the mode of use. Once you access an on-premises server or the cloud service, the editors are available along with the other services.

The editors work closely with the other modules, so that documents can be called up from the file manager without having to open a separate application. The editors' modern user interface is eye-catching (Figure 3); it is based on current design concepts with ribbon structure, tabs, and vertical toolbars at the edges of the screen, but it uses a syntax that takes some getting used to.

It is advisable to get to know the individual functions required before going live. The online editors in OnlyOffice come with the typical functions for daily office use, but have fewer options than a standard office package like OpenOffice or LibreOffice offers.

In the individual program windows, the *Collaboration* tab bundles the teamwork functions. In the matching buttonbar, you can define the access rights for the active document, and there is a comment function. The software can also track changes, including the ability to jump to the next change, for faster navigation in large documents.

After creating a new document and granting other users the appropriate access rights, the main editor can accept or reject their edits. To further help track modifications to files, OnlyOffice offers a comparison function in the *Collaboration* tab and also keeps a version history.

An integrated instant messenger, under the *Collaboration* tab, rounds off the functions for collaborative work. When you activate the software, a vertical bar appears on the left side of the screen, showing the message history.

There are two editing modes, *Fast* and *Strict*. By default, OnlyOffice uses the *Fast* option, which saves all changes made by participating editors in real time. The *Strict* option requires the main user to manually press *Save* to store the document and synchronize modifications.

A variety of plugins extend the functionality of the OnlyOffice package. OnlyOffice comes with some of these add-ons, which you can access under the *Plugins* tab. Plugins include an OCR module based on the Tesseract software for automated text recognition from images, a WordPress module for publishing documents from online editors, and a translation plugin based on the Russian Yandex engine.

Figure 3: OnlyOffice brings a very modern-looking user interface to the desktop via the web browser.
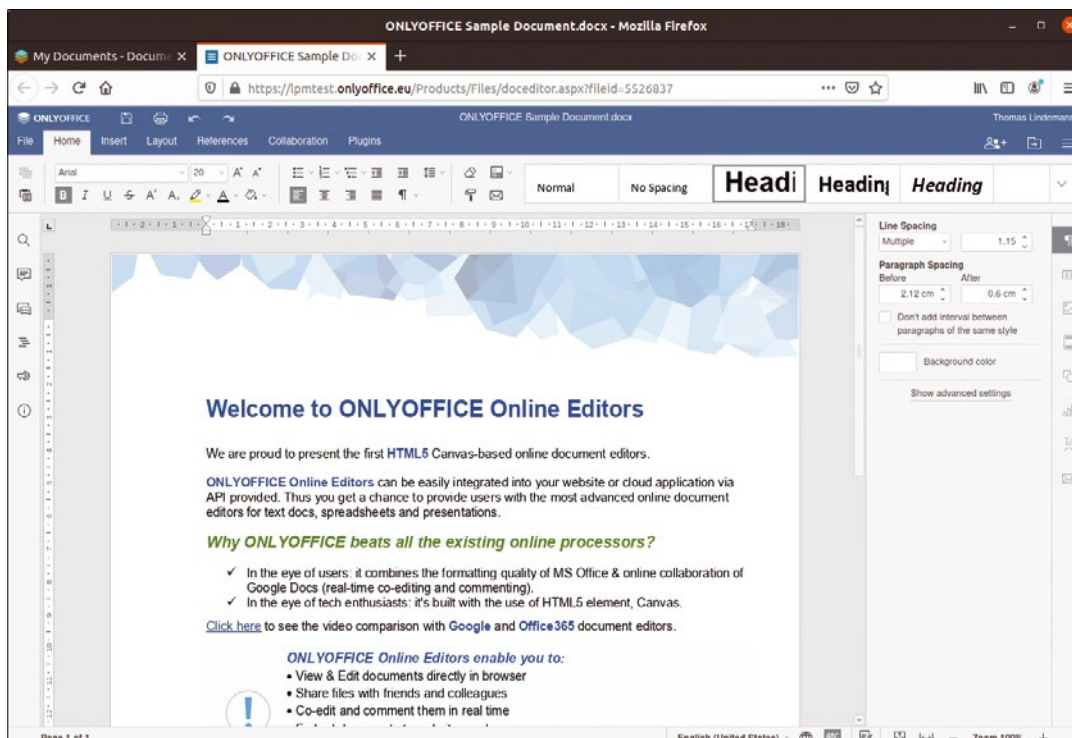
In addition, there is a tool for adjusting photos, one plugin each for managing and integrating code snippets from various programming and scripting languages, and an extension for managing macros. Another plugin supports the integration of YouTube videos into a document.

OnlyOffice can handle multiple file formats, but does not offer the flexibility of locally installed office suites. In addition to ODF, which is mainly used by LibreOffice and OpenOffice, the newer Open XML formats by Microsoft and PDF (read-only) are available. For web applications, the suite also supports HTML.

In our lab, however, converting between the formats showed the same issues that we saw with Collabora: To ensure a coherent layout, the same fonts, including all aspects and sizes, must be available on the source and target systems. Otherwise, you will have to manually edit the documents.

To better adapt OnlyOffice to your individual office needs, you can enable third-party extensions already included in the system. You can access these after logging in to the online portal; click on the *Select* box in the top left corner and then select *Services* from the drop-down menu.

The portal now displays a list of third-party apps. Many of them are integrated and activated in OnlyOffice by default. These are usually cloud services or providers for two-factor authentication. To unlock one of the apps, click on

the slider to the right of the extension. The program then opens a dialog in which you enter your access data for the service. Please note that the inactive services are generally commercial offerings for which you must have an account (Figure 4).

Desktop Editors

The installable OnlyOffice Desktop Editors [7] version is a useful alternative, especially where users are not allowed to work with online editors. The suite offers a very similar interface to the online editors, which minimizes the training overhead.

Like their online counterparts, the Desktop Editors contain a word processor, a spreadsheet, and a presentation program. The individual modules also contain a *Collaboration* tab. A cloud connection can be established from the start window, with options for Nextcloud, ownCloud, and the OnlyOffice cloud.

This variety of connections enables an exchange of data and collaborative work even with the locally installed version; ideally you would be hosting the Nextcloud and ownCloud environments on-premises. The desktop version offers the same plugin-based convenience features as the online editors.

Unlike the online versions, the installed solution supports encryption of documents. This option is available in all three modules and lets you create a password that is required to open the document.

Figure 4: The functionality of OnlyOffice can be extended by various apps, for example to provide connectivity to other systems.
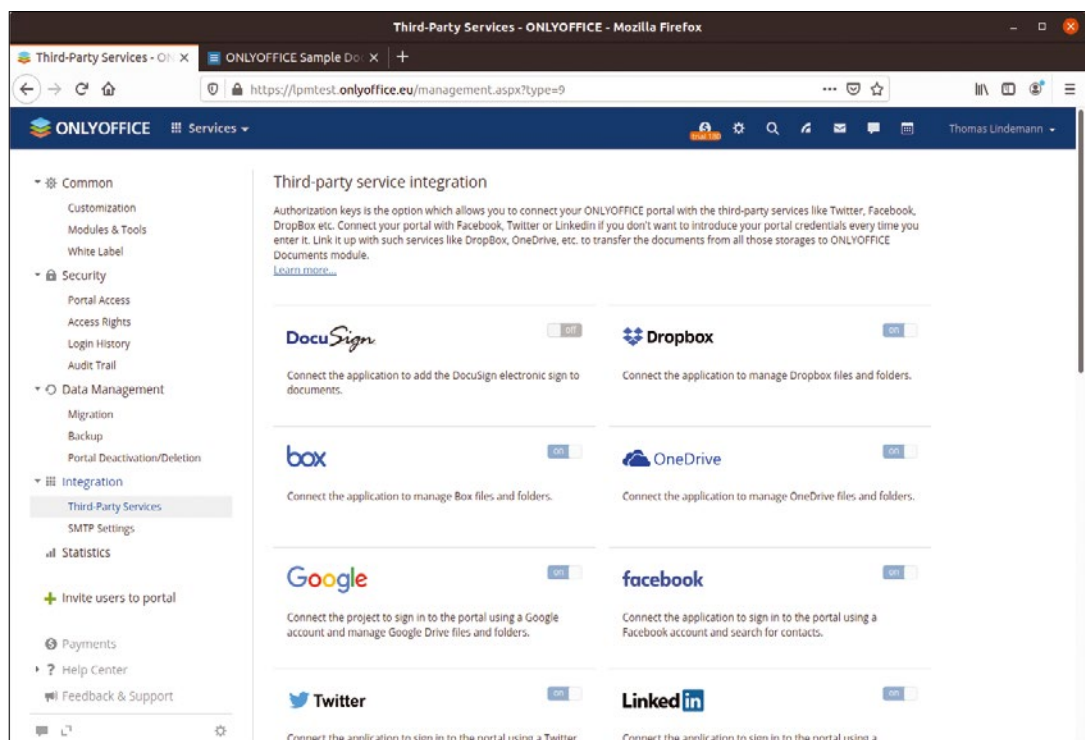
Conclusions

Collabora and OnlyOffice have many functions that will support collaborative work and do not pose users any problems for standard correspondence, conventional body text, and even more demanding documents with embedded graphics and images. Provided that the correct fonts are available, documents can also largely be converted without any problems. The back ends are similar. Both support the common cloud platforms as well as groupware packages, in which they can be integrated as additional applications if required.

The differences are in the details. While Collabora draws on LibreOffice's enormous feature set, OnlyOffice focuses more strongly on efficient editing of office documents and therefore offers a number of less common functions. Both packages are very flexible when it comes to platform independence ranging from in-house desktop apps, through cloud-based web applications, to desktop editors and apps for mobile devices. ▪▪▪

Info

[1]   Collabora Productivity:
      https://www.collaboraoffice.com

[2]   Collabora Office pricing models: *https://www.collaboraoffice.com/subscriptions/*

[3]   EGroupware: *https://www.egroupware.org*

[4]   Integration of Collabora in EGroupware:
      https://www.egroupware.org/en/collabora-online/

[5]   OnlyOffice: *https://www.onlyoffice.com*

[6]   Overview of OnlyOffice online editors:
      https://www.onlyoffice.com/office-suite.aspx

[7]   OnlyOffice Desktop Editors:
      https://www.onlyoffice.com/desktop.aspx

View a remote desktop with UltraScreen
Tech Support

UltraScreen lets you view the screen of a remote computer without a central server or paying for an account. It's a good option to have when someone calls you for computer help. BY CHRISTOPH LANGNER

When a printer simply won't print, when something has gone wrong with the package manager, or when other computer issues arise, it helps to be able to call on a friend, co-worker, or relative for help. However, you might not be able to meet in person, and the phone isn't ideal for this type of situation – it requires a lot of patience at both ends of the line to describe the problems.

Professional administrators, as well as many computer-savvy sons and daughters, instead turn to remote desktop solutions such as TeamViewer [1] or AnyDesk [2]. With the help of these programs, a connection can be established between two computers over the Internet, and you can operate the remote computer as if you were sitting at its keyboard. This works even without a fast Internet connection – modern compression methods keep the required data transfer rate to a minimum.

UltraScreen

Both TeamViewer and AnyDesk offer free options for personal use and are available for all common operating systems. However, the program vendors take great care to ensure that the user does not overstress their generosity. If there is an Exchange server on the network, or if the program is used conspicuously often and over long periods, they ask the user to purchase a commercial license.

Free alternatives like the classic VNC have the disadvantage that it is not so easy to route a connection securely through the Internet. Usually at least one of the participants will need to forward ports from the Internet router to the corresponding computer. This is where UltraScreen [3] steps in as a good alternative. The open source program does not require licensing or any configuration on the router side.

Installation

Like many young applications, you will search in vain for UltraScreen in the package sources of the popular distributions. Only Arch Linux has an entry in the Arch User Repository (AUR) in the form of *ultrascreen-git*. With the help of the AUR helper `yay`, the program can be quickly built and imported into the system (Listing 1, line 2). Alternatively, you can use the graphical package management front end Pamac developed by Manjaro, which can also be used on Arch Linux.

However, the developers also offer UltraScreen as a statically built version in the form of a ZIP archive and an AppImage. You only have to unpack the ZIP and then execute the binary `ultrascreen`, which is contained in the `linux-unpacked/` subdirectory (Listing 1, lines 4 and 5). Then make the AppImage file executable using `chmod` or a file manager and call the executable directly (Listing 1, lines 7 and 8). The advantage that AppImages offers is that the application can also be installed on the system this way and called up via the Application menu.

Making the Connection

Immediately after starting, UltraScreen displays just a white window with a *Login* button and a prompt to enter a nickname. The choice of the name is completely up to you; no account will be created on a server. However, the name can only contain letters and numbers, not even spaces are allowed. After logging in, UltraScreen then loads the actual application window (Figure 1).

Listing 1: Installation

```
01 ### Installation via AUR
02 $ yay -S ultrascreen-git
03 ### Unpack and call ZIP archive
04 $ unzip ultrascreen-v1.0-linux.zip
05 $ linux-unpacked/ultrascreen
06 ### Run AppImage
07 $ chmod +x ultrascreen-v1.0.AppImage
08 $ ./ultrascreen-v1.0.AppImage
```
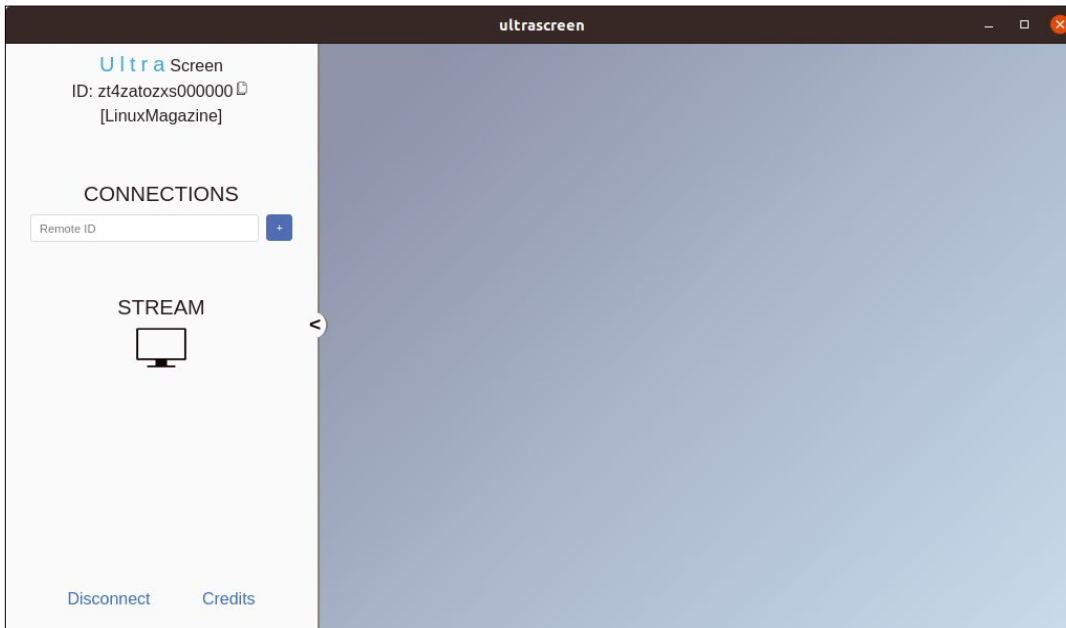
Figure 1: The UltraScreen ID is displayed in the top left-hand corner of the application window.

Below the program name in the left sidebar you will find the ID number of your UltraScreen instance in the style of *719[…]00000*. UltraScreen creates a new ID every time you log in; there is no way to get a fixed ID. Below the ID is the nickname you chose previously.

If you're trying to connect with someone who needs computer help, you need the UltraScreen ID of that person. This can be communicated either by email or chat, or even by phone if needed. Use the icon to the right of the ID to copy the ID to the clipboard and then press Ctrl+V to paste it into a chat or email program.

The person who wants your help has to tap on the large screen icon labeled *STREAM*. UltraScreen then shows them a selection: They can transfer either the complete desktop (*Entire screen*) or only individual applications (Figure 2). This is a little unfortunate for users with two or more monitors: As of now, there is no way to stream only the content of one screen, which unnecessarily inflates the volume of data to be transferred in multi-display setups. After selecting the content, UltraScreen displays a preview of the Internet screencast in the application window.

Figure 2: UltraScreen lets you stream the complete desktop of the computer or, if so desired, transfer only the content of single windows.
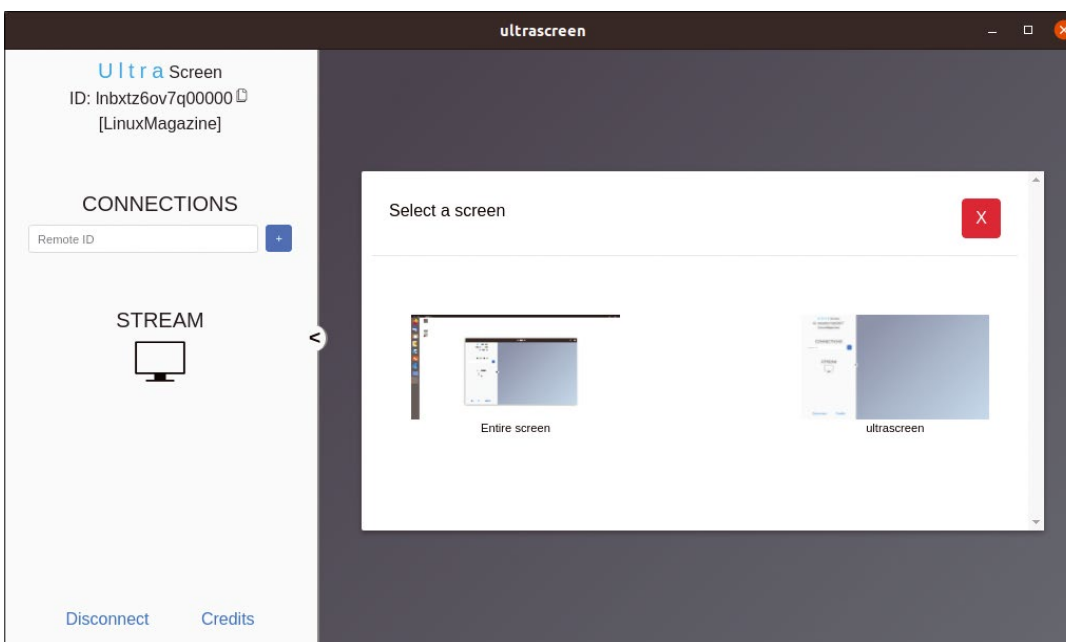
You now need to enter the remote ID of the other person below *CONNECTIONS*. Pressing the plus icon tells UltraScreen to save the connection and paste it below the input field. However, the entry will be gone the next time you start the program, because it changes in every new session.

As soon as the connection is established, you will see the desktop or the selected window that you need to investigate. UltraScreen only transmits the image. There is no return channel to remotely control the remote computer (Figure 3). The screencast remains active until one of the participants taps *Disconnect* at the bottom of the screen.

Wayland Workaround

Like many other screencast and screen recorder programs, UltraScreen fails if it encounters Wayland, the successor of the classic X server. On Gnome, for example, UltraScreen only transmits a black box. In order to be able to use the program anyway, the person seeking help has to log out of the Gnome desktop in this case and then select the *GNOME on Xorg* option via the gear menu in the login dialog. Doing so launches a Gnome session with a classic X server. To discover whether the desktop is running on Wayland you can quickly type the `echo $XDG_SESSION_TYPE` command.

Conclusions

In practice, UltraScreen proves to be a helpful tool. It does not necessarily require any installation work on the computer of the person seeking help, and there are variants for Linux and Windows, which makes the program helpful in many situations. In the test, the connection setup worked reliably, both with participants on home networks and with users on more restrictive enterprise networks, and did not require any changes to the network configuration of the router, nor did it require the assistance of an administrator.

However, UltraScreen does not yet come close to the feature scope of commercial remote desktop solutions such as TeamViewer;. the transferred desktop cannot be remotely controlled, nor is there any way to communicate with the person seeking help via the application or to transfer files between computers. Nevertheless, the feature set offered will do the job in many situations. Just having the problem right in front of your eyes often eliminates the biggest difficulties. ■■■

Info

[1] TeamViewer: *https://www.teamviewer.com*

[2] AnyDesk: *https://anydesk.com*

[3] UltraScreen: *https://github.com/w3yden/ultrascreen*

Figure 3: In contrast to commercial remote desktop applications, UltraScreen only allows you to view the desktop of the counterpart. There is no remote control.
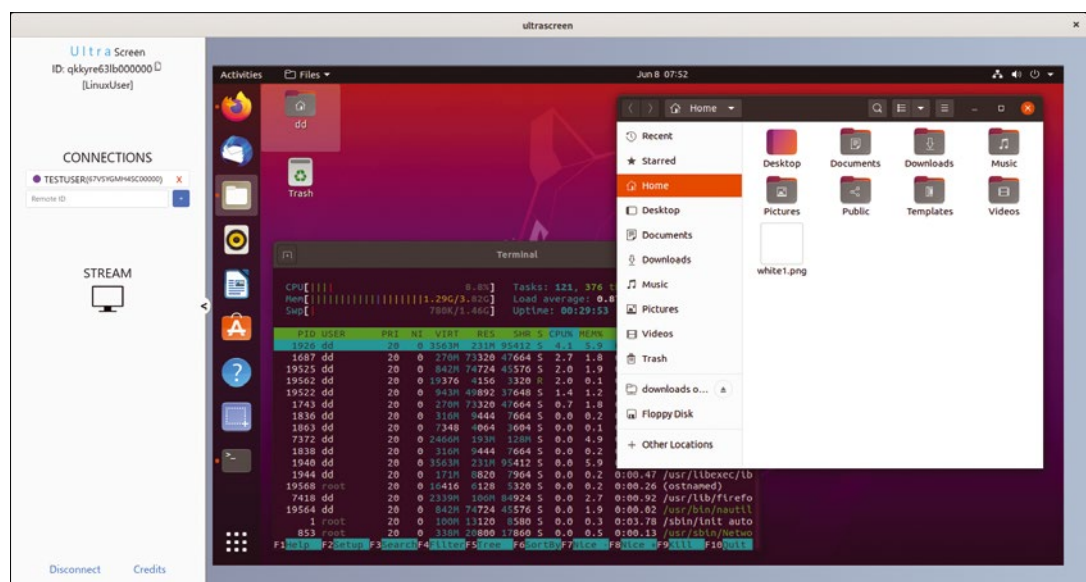
LINUX
NEWSSTAND

Linux Magazine is your guide to the world of Linux. Monthly issues are packed with advanced technical articles and tutorials you won't find anywhere else. Explore our full catalog of back issues for specific topics or to complete your collection.

#237/August 2020
Webcams and Linux

This month we explore webcams, screencasting, and a cool teleprompter tool. We also look at PHP Building Blocks, Guacamole the clientless remote access tool based on HTML5, and a study of the handy MystiQ Audio/Video conversion tool.

On the DVD: Ubuntu Studio 20.04 and Kubuntu 20.04

#236/July 2020
Smarter Directories

The rigid structure of nested files and directories used on computer systems around the world was created more than 60 years ago, and experts believe we can do better. This month, you'll learn about some scripts for semantic file tagging in Linux.

On the DVD: Fedora Workstation 32 and Ubuntu "Focal Fossa" Desktop 20.04 LTS

#235/June 2020
What's New in Systemd

Systemd is a mystery that keeps on giving. Now a new feature of the leading Linux init system will change the way you think about user home directories. This month we take a closer look at systemd-homed.

On the DVD: Knoppix 8.6.1 and OpenMandriva Lx Plasma 4.1

#234/May 2020
Edge Computing

The Edge is a popular buzz word in high-tech news, but what does it mean really? We introduce you to an exciting new technology that could be changing the way we think about the cloud.

On the DVD: Manjaro 19.02 Gnome Edition and SystemRescueCd 6.1

#233/April 2020
Stream to Your TV

The line between computers and television blurred long ago, but the new tools and new ideas keep coming. This month we highlight some innovative apps for multimedia in Linux, including Gnome Cast for TV, and the easy-to-use Serviio media server.

On the DVD: The Complete Raspberry Pi Geek Archive

#232/March 2020
Stop Ads

Browser-based ad-blockers are useful for controlling many types of pop-ups and banners, but they are less effective with ads built into applications. We look at a couple of alternative tools for blocking ads at the network level: Pi-hole and Privoxy.

On the DVD: GParted 1.0.0 and Kali Linux 2019.4

FEATURED EVENTS

Users, developers, and vendors meet at Linux events around the world. We at *Linux Magazine* are proud to sponsor the Featured Events shown here.

For other events near you, check our extensive events calendar online at *https://www.linux-magazine.com/events.*

If you know of another Linux event you would like us to add to our calendar, please send a message with all the details to *events@linux-magazine.com*.

NOTICE

Be sure to check the event website before booking any travel, as many events are being canceled or converted to virtual events due to the effects of COVID-19.

Storage Developer Conference

Date: September 22-23, 2020

Location: Virtual Event

Website: *https://bit.ly/SDC-2020-virtual*

SDC 2020 is a cost-effective way to acquire training in a host of different areas through tutorials, sessions, Keynote speakers, and the opportunity to participate in co-located plugfests for the storage development community.

openSUSE + LibreOffice Conference

Date: October 15-17, 2020

Location: Virtual Event

Website: *https://events.opensuse.org/*

The openSUSE and LibreOffice Projects are celebrating 10 years of the LibreOffice Project and 15 years of the openSUSE Project with a combined virtual conference. Members of the open source community will come together to learn, teach, and share their passion for open source.

Events

Event	Date	Location	Website
Cloud Native Security Day	August 17	Virtual Event	https://bit.ly/Cloud-Native-Security-Day
KubeCon + CloudNativeCon	August 17-20	Virtual Event	https://bit.ly/kubecon-cloudnativecon
DevOpsCon	August 31-Sept 3	London, United Kingdom	https://devopscon.io/london/
Open Source Summit Japan	September 15-16	Tokyo, Japan	https://bit.ly/open-source-japan
Storage Developer Conference	September 22-23	Santa Clara, California	https://www.snia.org/events/storage-developer
Drupal GovCon 2020	September 24-25	Virtual Event	https://www.drupalgovcon.org/
Open Networking & Edge Summit North America	September 28-29	Virtual Event	https://bit.ly/Edge-summit
Smart Grid Cybersecurity 2020	October 6-8	Berlin, Germany	https://bit.ly/smart-grid-cybersecurity
openSUSE + LibreOffice Conference	October 15-17	Everywhere	https://events.opensuse.org/conferences/oSLO
All Things Open	October 19-20	Virtual Event	https://2020.allthingsopen.org/
Open Source Summit Europe	October 26-28	Dublin, Ireland	https://bit.ly/open-source-europe
KVM Forum	October 28-30	Dublin, Ireland	https://bit.ly/KVM-forum
Cloud Foundry Summit Europe	October 29	Dublin, Ireland	https://bit.ly/cloud-foundry-summit
Linux Security Summit	October 29-30	Dublin, Ireland	https://bit.ly/Linux-Security-Europe
TechWeek Franfurt 2020	November 4-5	Frankfurt, Germany	https://www.techweekfrankfurt.de/
SC20	Novemer 15-20	Atlant, Georgia	https://sc20.supercomputing.org/
KubeCon + CloudNativeCon North America	November 17-20	Virtual Experience	https://bit.ly/KubeCon-North-America

Images © Alex White, 123RF.com

CALL FOR PAPERS

We are always looking for good articles on Linux and the tools of the Linux environment. Although we will consider any topic, the following themes are of special interest:

- System administration
- Useful tips and tools
- Security, both news and techniques
- Product reviews, especially from real-world experience
- Community news and projects

If you have an idea, send a proposal with an outline, an estimate of the length, a description of your background, and contact information to *edit@linux-magazine.com*.

The technical level of the article should be consistent with what you normally read in *Linux Magazine*. Remember that *Linux Magazine* is read in many countries, and your article may be translated into one of our sister publications. Therefore, it is best to avoid using slang and idioms that might not be understood by all readers.

Be careful when referring to dates or events in the future. Many weeks could pass between your manuscript submission and the final copy reaching the reader's hands. When submitting proposals or manuscripts, please use a subject line in your email message that helps us identify your message as an article proposal. Screenshots and other supporting materials are always welcome.

Additional information is available at:
http://www.linux-magazine.com/contact/write_for_us.

NOW PRINTED ON recycled paper from 100% post-consumer waste; no chlorine bleach is used in the production process.

Authors

Erik Bärwaldt	86
Chris Binnie	44
Zack Brown	12
Bruce Byfield	6, 34, 38
Joe Casad	3
Mark Crutch	67
Jon "maddog" Hall	68
Charly Kühnast	41
Christoph Langner	74, 92
James Mawson	69
Vincent Mealing	67
Paul Menzel	20
Martin Mohr	62
Graham Morrison	78
Rob Reilly	56
Jörg Schorn	24
Mayank Sharma	16
Ferdinand Thommes	50
Alexander Tolstoy	30
Jack Wallen	8

Issue 239 / October 2020

Build Your Own IRC Bot

Businesses, non-profits, political campaigns, and any other organizations depend on chat to support their online communities. Next month we'll show you how to automate your chat presence with a homebuilt IRC bot, and we'll give you some etiquette tips for deploying your chat bot responsibly in an active user space.
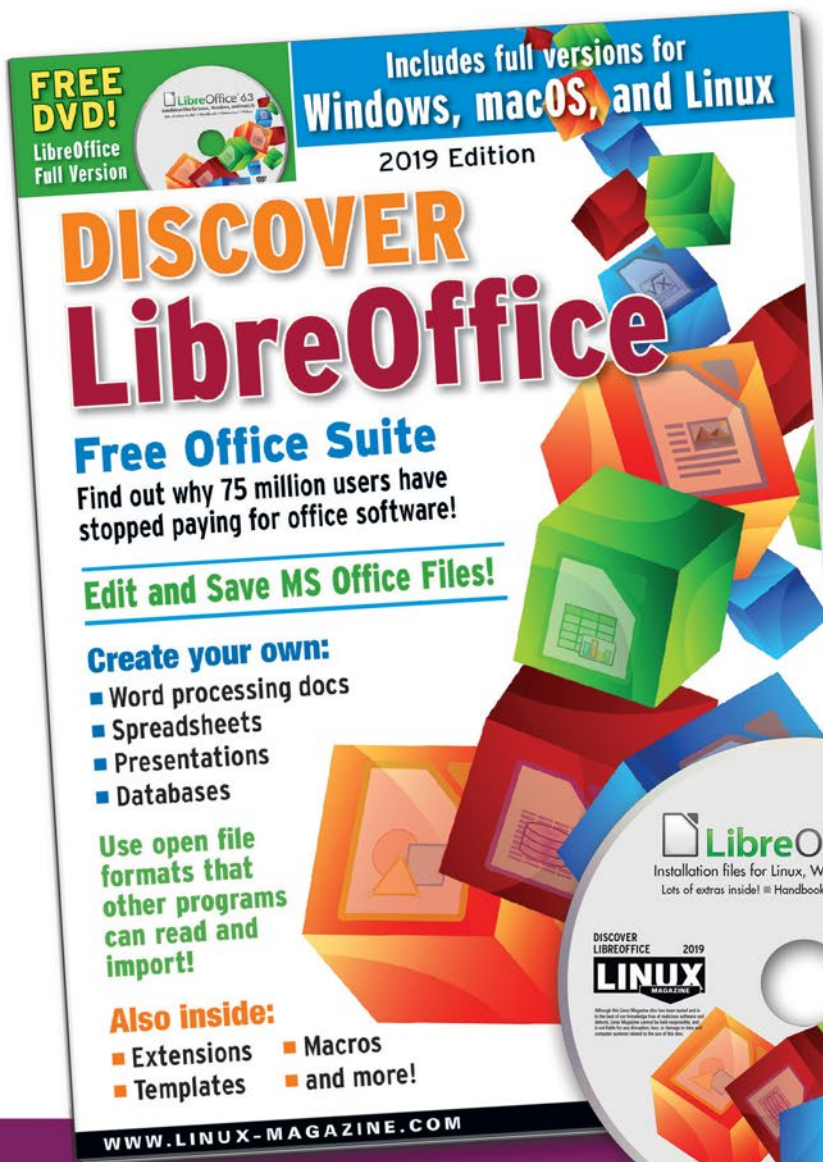
Lead Image © alexandersikov, 123RF.com

Preview Newsletter

The Linux Magazine Preview is a monthly email newsletter that gives you a sneak peek at the next issue, including links to articles posted online.

Sign up at: *https://bit.ly/Linux-Update*